TUGAS AKHIR – K141502

RANCANG BANGUN APLIKASI FINDING-TUTOR BERBASIS ANDROID DAN PENENTUAN PRIORITAS SELEKSI MURID

RISKA ADHITA
NRP 5113100079

Dosen Pembimbing
Dr. tech. Ir. Raden Venantius Hari Ginardi, M.Sc.

DEPARTEMEN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2017

TUGAS AKHIR – K141502

RANCANG BANGUN APLIKASI FINDING-TUTOR BERBASIS ANDROID DAN PENENTUAN PRIORITAS SELEKSI MURID

Riska Adhita
NRP 5113100079

Dosen Pembimbing 1
Dr. tech. Ir. Raden Venantius Hari Ginardi, M.Sc.

DEPARTEMEN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2017

[Halaman ini sengaja dikosongkan]

FINAL PROJECT – K141502

SOFTWARE DESIGN OF FINDING-TUTOR ANDROID APPLICATION AND DETERMINING PRIORITY OF STUDENT CLIENTS

RISKA ADHITA
NRP 5113100079

Supervisor 1
Dr. tech. Ir. Raden Venantius Hari Ginardi, M.Sc.

INFORMATICS DEPARTMENT
Information Technology Faculty
Sepuluh Nopember Institute of Technology
Surabaya 2017

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

RANCANG BANGUN APLIKASI FINDING-TUTOR BERBASIS ANDROID DAN PENENTUAN PRIORITAS SELEKSI MURID

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Manajemen Informasi
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh :
RISKA ADHITA
NRP : 5113 100 079

Disetujui oleh Dosen Pembimbing Tugas Akhir :

Dr. tech. Ir. Raden Venantius Hari Ginardi, M.Sc. .....
NIP: 196505181992031003 (pembimbing 1)

SURABAYA
JUNI 2017

[Halaman ini sengaja dikosongkan]

RANCANG BANGUN APLIKASI FINDING-TUTOR BERBASIS ANDROID DAN PENENTUAN PRIORITAS SELEKSI MURID

Nama Mahasiswa	:	Riska Adhita
NRP	:	5113 100 079
Jurusan	:	Teknik Informatika FTIf-ITS
Dosen Pembimbing 1	:	Dr. tech. Ir. Raden Venantius Hari Ginardi, M.Sc.

ABSTRAKSI

Surabaya merupakan kota terbesar kedua di Indonesia, dengan predikat itu, Surabaya memiliki jumlah sekolah yang cukup banyak. Untuk mendukung proses pembelajaran, para orang tua murid dan juga murid sendiri sering kali mencari tutor atau guru les.

Sering kali tutor atau guru les yang didapat tidak sesuai dengan kriteria dari pencari tutor, dan juga tutor atau guru les sering mendapatkan murid yang tidak termasuk pada prioritas dari tutor atau guru les. Saat ini masih belum ada aplikasi untuk mendapatkan murid yang termasuk pada prioritas dari tutor dan juga untuk mendapatkan tutor yang sesuai dengan kriteria dari murid.

Pada tugas akhir ini ingin menawarkan penyelesaian masalah yang ada. Aplikasi ini akan memberikan murid yang termasuk pada prioritas tutor dengan parameter kriteria murid yang diinginkan tutor dan juga memberikan tutor yang sesuai dengan keinginan dari murid. Penentuan prioritas murid bagi tutor ditentukan oleh kecocokan yang dilihat dari kriteria yang diberikan murid dengan keadaan dari tutor, mulai dari pemilihan jenis kelamin, usia, pelajaran, dan jarak yang diinginkan murid maupun tutor.

Aplikasi ini menggunakan perangkat bergerak yang dapat menunjang mobilitas para murid, serta para tutor atau guru les. Selain itu, pada tugas akhir ini akan menggunakan *Google API* untuk menunjang pencarian tutor dan mendapatkan prioritas murid.

Pada tahap pengujian, akan dilakukan dengan transaksi pencarian tutor dan pencarian murid. Pengujian dilakukan untuk mengetahui keberhasilan aplikasi.

Kata kunci: Google API, Pencarian Tutor, Perangkat Bergerak

SOFTWARE DESIGN FINDING-TUTOR ANDROID BASED APPLICATION AND DETERMINING PRIORITY OF STUDENT SELECTION

Student Name : Riska Adhita
Student ID : 5113 100 079
Major : Informatics Department FTIf-ITS
Advisor 1 : Dr. tech. Ir. Raden Venantius Hari Ginardi, M.Sc.

ABSTRACTION

Surabaya now is the second biggest city in Indonesia. Because of that predicate, Surabaya has many schools started from primary school, junior high school, until senior high school. In order to provide learning process, student's parents with the student itself have to find a skillful teacher.

However, many teachers frequently don't find suitable transaction from the student which is far different from their both criteria and ability. Otherwise, not only teacher experienced the condition, but also the students feel the same way. As the time flew, nowadays, there is no application to provide teachers in getting student which is included as their criteria and allow student to get a skillful teacher as well.

On this Theses, this book has offered a solution for teaching and learning process. By creating Finding – Tutor application with android base, Finding – Tutor Application will provide students which are included as teacher's priority by adding student criteria parameter that has been inputted by the teachers first. Moreover, students also can be helped from this application by allowing students in getting suitable teacher in order to help them in learning process. Student priority determination that is inputted by teachers, is chosen by student criteria matching combined with teachers condition, started from

gender preference, age preference, lesson preference, until distance for the transaction between teacher and student.

This application uses android system in order to provide the mobility of the student and the teacher as well. Besides, on this theses, Google API is used by Finding – Tutor application to support teacher's searching and get student's priority.

On the testing phase, transaction which involved finding teachers is used by the application. The goal of the testing phase is to determine the successful rate of the application

Keyword : Finding Tutor, Google API , Mobile Device

KATA PENGANTAR

Puji syukur kepada Tuhan Yang Maha Esa atas segala karunia dan rahmat-Nya penulis dapat menyelesaikan Tugas Akhir yang berjudul:

RANCANG BANGUN APLIKASI FINDING-TUTOR BERBASIS ANDROID DAN PENENTUAN PRIORITAS SELEKSI MURID

Melalui lembar ini, penulis ingin menyampaikan ucapan terimakasih dan penghormatan yang sebesar-besarnya kepada:

1. Mama, kakak dan keluarga besar yang selalu memberikan dukungan penuh untuk menyelesaikan tugas akhir ini.
2. Bapak Dr. tech. Ir. Raden Venantius Hari Ginardi, M.Sc selaku dosen pembimbing I yang telah banyak menyampaikan ilmu dan bimbingan yang tak ternilai harganya bagi penulis.
3. Bapak, Ibu dosen Departemen Teknik Informatika ITS yang telah banyak memberikan ilmu dan bimbingan yang tak ternilai harganya bagi penulis.
4. Rekan, teman dan sahabat administrator Laboratorium Manajemen Informasi, Naufal, Nanang, Lino, Anne, Nay, Kania, Haidar, Adit, Huda, dan Unggul, yang selalu mewarnai hari-hari penulis di laboratorium.
5. Rei, Demsy, Luffy, Nyoman, Uul, Aldi, Dimas, Izar, Nanda, Razi, Cayza, Afif, Fajar, Ghulam, John, Nindy, Zaza yang seringkali memberikan hiburan dengan bermain bersama
6. Teman-teman “Sahabat Alpro” yang sering memberikan wejangan-wejangan serta memberikan dukungan moril kepada penulis.
7. Teman-teman TC angkatan 2013 yang telah membantu dan berbagi segala informasi kepada penulis selama masa perkuliahan.

8. Rifqi dan Dhita selaku anggota kelompok Carikos yang selalu memberikan saran kepada penulis selama pengerjaan tugas akhir ini.
9. Dhea dan Nyoman selaku teman sekelompok TA yang telah berjuang bersama untuk menyelesaikan tugas akhir ini.
10. Alfonza Nugrahaning Kristi, S.Psi yang selalu menemani, membantu, dan memberikan "semangat" kepada penulis hingga saat ini.
11. Serta pihak-pihak lain yang namanya tidak dapat penulis sebutkan satu per satu.

Bagaimanapun juga penulis telah berusaha sebaik-baiknya dalam menyusun Tugas Akhir ini, namun penulis mohon maaf apabila terdapat kekurangan yang penulis lakukan. Kritik dan saran yang membangun dapat disampaikan sebagai bahan perbaikan selanjutnya.

Surabaya, Juni 2017

Riska Adhita

DAFTAR ISI

[Halaman ini sengaja dikosongkan]

DAFTAR GAMBAR

[Halaman ini sengaja dikosongkan]

DAFTAR TABEL

DAFTAR KODE SUMBER

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

Pada bab ini akan dipaparkan mengenai garis besar Tugas Akhir yang meliputi latar belakang, tujuan, rumusan dan batasan permasalahan, metodologi pembuatan tugas akhir, dan sistematika penulisan.

1.1. Latar Belakang

Surabaya merupakan kota terbesar kedua di Indonesia, dengan jumlah sekolah sebanyak 1.510 sekolah [1]. Apabila melihat dari jumlah tersebut, terdapat berbagai macam peluang bisnis, salah satunya adalah bisnis menjadi guru les atau tutor. Sebagai pelajar, kebutuhan untuk memiliki tutor atau guru les untuk menunjang pendidikan yang ditempuh. Namun kenyataannya pelajar atau pencari tutor masih kesulitan menemukan tutor atau guru les, hal ini dapat dilihat bahwa masih belum ada media yang pasti bagi pelajar atau pencari tutor untuk mencari guru les atau tutor. Hal yang sama juga terjadi pada pihak tutor atau guru les, yaitu tidak menjadi prioritas pertama untuk mendapatkan murid karena terbentur oleh kriteria-kriteria dari masing-masing pelajar atau pencari tutor.

Untuk mencari tutor atau guru les yang sesuai dengan keinginan atau kebutuhan, sering kali para pencari tutor memiliki kriteria guru les atau tutor tersebut, diantaranya adalah lokasi pelaksanaan tutor atau les, jenis kelamin, usia tutor atau guru les, biaya yang diinginkan oleh tutor atau guru les. Begitu juga sebaliknya, tutor atau guru les ingin mendapatkan murid yang masuk pada prioritas dari tutor atau guru les dengan melihat kondisi atau kriteria dari pentutor atau guru les, mulai jarak tempuh tempat les atau tutor dilaksanakan hingga biaya yang harus dikeluarkan oleh pelajar tersebut. Berdasarkan kriteria yang dicantumkan oleh pencari tutor maupun tutor atau guru les akan menemukan kecocokan. Melalui kecocokan tersebut, pelajar atau pencari tutor akan mendapatkan pentutor atau guru les sesuai

dengan kriteria yang diinginkan, serta sebaliknya, pentutor atau guru les akan mendapatkan daftar prioritas pelajar atau pencari tutor yang sesuai dengan kebutuhannya.

Melihat kebutuhan yang ada serta pesatnya teknologi, hadirlah *Finding-Tutor*. *Finding-Tutor* adalah aplikasi berbasis *mobile android* yang mengimplementasikan semua kebutuhan di atas, mulai dari penyediaan informasi tutor atau guru les yang ada, pencarian tutor atau guru les sesuai dengan kriteria yang diinginkan, serta pencocokan yang berujung pada pembuatan prioritas bagi tutor atau guru les untuk mendapatkan murid yang sesuai dengan kriteria yang diinginkan.

1.2. Tujuan

Tujuan pembuatan tugas akhir ini adalah:

1. Memberikan kriteria pentutor kepada pencari tutor sesuai dengan kebiasaan atau *history* pencari tutor.
2. Memberikan tutor atau guru les yang sesuai dengan kriteria dari pencari tutor.
3. Memberikan informasi pencarian tutor kepada tutor atau guru les berdasarkan prioritas yang dilihat dari kondisi tutor atau guru les.

1.3. Rumusan Permasalahan

Rumusan masalah yang diangkat dalam tugas akhir ini antara lain:

1. Bagaimana memberikan kriteria tutor secara otomatis kepada pencari tutor?
2. Bagaimana aplikasi dapat menemukan tutor atau guru les yang sesuai dengan kriteria pencari tutor?
3. Bagaimana aplikasi dapat menampilkan informasi mengenai pencarian tutor berdasarkan prioritas yang dilihat dari kondisi tutor atau guru les?

1.4. Batasan Permasalahan

Permasalahan yang dibahas dalam tugas akhir ini memiliki beberapa batasan, antara lain:

1. Jangkauan aplikasi pada daerah Surabaya
2. Penentuan prioritas hanya dilihat dari tingkat kecocokan antara kriteria pencari tutor dengan keadaan tutor atau guru les.
3. Penentuan kecocokan dilihat dari jarak, jenis kelamin, usia, biaya, materi, dan ketersediaan hari dari masing-masing *input* antara pencari tutor dan tutor atau guru les.
4. Lokasi tutor diambil dari tempat tinggal dari tutor atau guru les.
5. Aplikasi dapat digunakan dengan perangkat Android veris 5.0 keatas.
6. Servis peta yang digunakan adalah *Google Maps API.*

1.5. Metodologi

Langkah-langkah yang ditempuh dalam pengerjaan Tugas Akhir ini yaitu:

a. Penyebaran Kuesioner

Penyebaran kuesioner digunakan penulis sebagai tolok ukur, menurut masyarakat kriteria untuk mencari pelajar atau pencari tutor dan untuk mencari tutor atau guru les seperti apa. Kuesioner diberikan kepada mahaiswa dan mahasiswi serta orangtua yang berjumlah 30 responden. Pada kuesioner ini diberikan nilai prioritas satu sampai tiga, dengan nilai satu merupaka nilai prioritas paling kecil dan nilai tiga merupakan nilai priorita tertinggi. Dari kuesioner ini akan dijadikan tolok ukur bagi penulis untuk mengembangkan aplikasi. Detail kriteria dan hasil dari kuesioner dapat dilihat pada Tabel 1.1.

Tabel 1.1 Tabel Kriteria Tutor

No	Pertanyaan	Nilai Prioritas		
		1	2	3
1	Seberapa penting jarak tempuh dilaksanakannya tutor atau les?	3,3%	13,3%	83,3%
2	Seberapa penting jenis kelamin dari guru les atau tutor?	26,7%	36,7%	36,7%
3	Seberapa penting usia dari tutor atau guru les?	26,7%	36,7%	36,7%
4	Seberapa penting biaya tutor atau guru les?	6,7%	30%	63,3%

Kemudian dalam kuesioner terdapat pertanyaan terbuka untuk menampung saran yang berisi kriteria apalagi yang dapat digunakan untuk mencari pelajar atau pencari tutor dan juga untuk mencari tutor atau guru les, serta didapat waktu dilaksanakannya tutor atau les dan materi pembelajaran.

b. Studi literatur

Pada tahap ini akan dicari studi literatur yang relevan untuk dijadikan referensi dalam pengerjaan tugas akhir. Studi literatur dapat diambil dari buku, internet, ataupun materi dalam suatu mata kuliah yang berhubungan dengan metode yang akan digunakan.

c. Analisis dan desain perangkat lunak

Aktor dari aplikasi ini adalah semua orang, khususnya pada pencari tutor dan tutor. Fitur dari aplikasi ini antara lain pengguna dapat mencari seorang tutor sesuai dengan keinginan, pencari tutor mendapatkan informasi mengenai tutor dan pencari tutor dapat mengetahui informasi umum dari tutor atau

guru les serta pencari tutor akan mendapatkan rekomendasi tutor atau guru les dari kebiasaan melakukan pencarian tutor.

d. Implementasi perangkat lunak

Implementasi perangkat lunak ini dibangun dengan bahasa pemrograman java dan *database MySQL*. Selain itu target pengguna aplikasi adalah pengguna android versi 5.0 keatas.

e. Pengujian dan evaluasi

Pengujian akan dilakukan oleh lebih dari satu orang pencari tutor dan pentutor. Pencari tutor akan memasukkan beberapa kriteria mengenai tutor atau guru les, kemudian pentutor diminta untuk memilih satu pencari tutor. Pencari tutor sendiri akan melakukan pencarian tutor lebih dari satu kali untuk mengetahui rekomendasi tutor atau guru les.

1.6. Sistematika Penulisan

Buku Tugas Akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan Tugas Akhir ini. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku Tugas Akhir terdiri atas beberapa bagian seperti berikut ini.

Bab I Pendahuluan

Bab ini berisi latar belakang masalah, tujuan dan manfaat pembuatan Tugas Akhir, permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penyusunan Tugas Akhir.

Bab II Dasar Teori

Bab ini membahas beberapa teori penunjang yang berhubungan dengan pokok pembahasan dan mendasari pembuatan Tugas Akhir ini.

Bab III Analisis dan Perancangan Sistem

Bab ini membahas mengenai perancangan perangkat lunak. Perancangan perangkat lunak

meliputi perancangan data, arsitektur, proses dan perancangan antarmuka aplikasi.

Bab IV Implementasi

Bab ini berisi implementasi dari perancangan dan implementasi fitur-fitur penunjang aplikasi.

Bab V Pengujian dan Evaluasi

Bab ini membahas pengujian dengan metode kotak hitam (*black box testing*) untuk mengetahui aspek nilai fungsionalitas dari perangkat lunak dan nilai kegunaan yang dibuat dengan juga memperhatikan ketertarikan pada calon partisipan untuk menggunakan aplikasi ini.

Bab VI Kesimpulan dan Saran

Bab ini berisi kesimpulan dari hasil pengujian yang dilakukan. Bab ini membahas saran-saran untuk pengembangan sistem lebih lanjut.

Daftar Pustaka

Merupakan daftar referensi yang digunakan untuk mengembangkan Tugas Akhir.

BAB II
DASAR TEORI

Pada bab ini akan dibahas mengenai teori-teori yang menjadi dasar dari pembuatan Tugas Akhir.

2.1. Aplikasi Sejenis

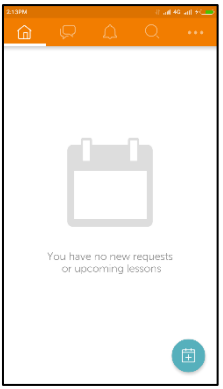
Sebelum merancang dan mengimplementasikan sistem, meninjau aplikasi serupa dilakukan guna menentukan standarisasi dalam aplikasi pencarian tutor. Aplikasi yang dipilih adalah *Chegg Tutor*. *Chegg Tutor* hanya mempertemukan tutor dengan murid melalui *online chat*. Tutor yang telah disediakan sudah dikategorikan menurut pelajaran yang telah disediakan oleh aplikasi.

Terdapat beberapa fitur yang ada pada aplikasi ini, yaitu:

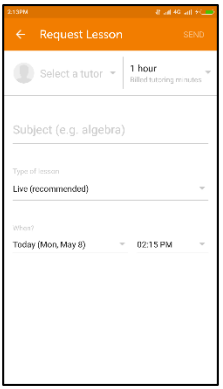
1. ***Request or Upcoming Lessons***

   Jika kita pengguna pertama dan belum melakukan *chat* dengan tutor yang telah disediakan, kita akan dialihkan menuju fitur ***Find Tutor by Subject*** berisi daftar pelajaran dan tutor yang menguasai pelajaran itu. Setelah itu kita diharuskan *chat* kepada tutor terkait. Setelah melakukan chat, kita baru bisa melakukan *request* pelajaran.

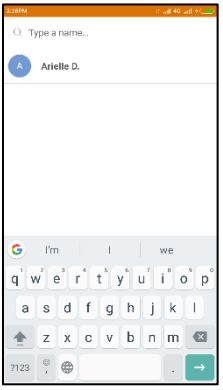
   Dalam melakukan *request* terdapat formulir yang wajib diisi oleh pencari tutor, pertama kita harus memilih tutor terlebih dahulu. Namun kita hanya bisa memesan tutor yang pernah pengguna *chat*. Terdapat pula daftar *subject* berisi materi pembelajaran yang tutor itu kuasai saja, pencari tutor tidak bisa menambahkan materi yang diinginkan. Terdapat dua pilihan pembelajaran, *live* atau *written*. *Live* disediakan bagi pencari tutor yang sedang membutuhkan tutor pada saat itu juga dengan menggunakan *text*, *video*, dan *audio chat*. Sedangkan *written* disediakan bagi pencari tutor yang telah melakukan tugasnya, dan tutor akan melakukan pemeriksaan pada tugas yang kita kirim. Dalam formulir juga berisi lama pembelajaran serta kapan akan melakukan pembelajaran.



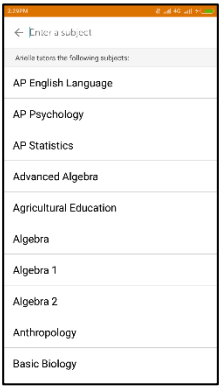
Gambar 2.1 *Screenshot* Halaman Utama Fitur Request or Upcoming Lessons



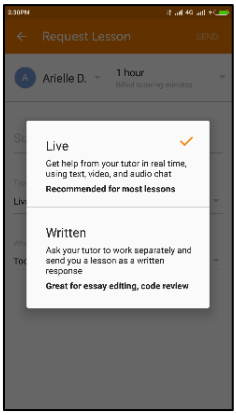
Gambar 2.2 *Screenshot* Halaman Formulir



Gambar 2.3 *Screenshot* Daftar Tutor yang Sudah Terhubung



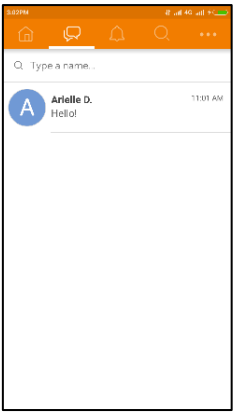
Gambar 2.4 *Screenshot* Daftar Pelajaran yang Dikuasai Tutor



Gambar 2.5 *Screenshot* Tipe Pembelajaran

2. ***Connect With Tutor***

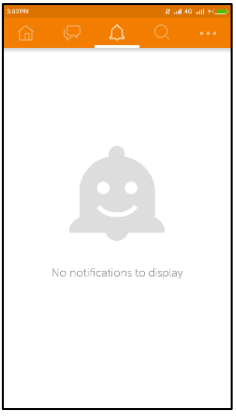
Pada fitur ini kita dapat melakukan *chat* dengan tutor yang pernah pencari tutor *chat*. Jika menuju halaman ini, pertama kita akan diberikan daftar tutor yang pernah kita *chat*. Sama seperti fitur ***Request or Upcoming Lessons***, jika belum pernah melakukan *chat* akan diarahkan ke daftar tutor berdasarkan keahlian pembelajaran.



Gambar 2.6 *Screenshot* Daftar Tutor yang Sudah Terhubung

3. ***Notification***

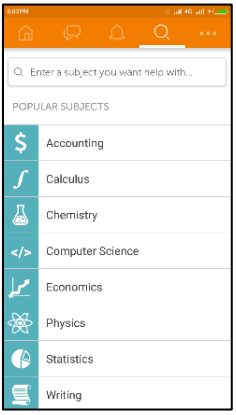
Pada fitur ini masih belum diketahui fungsinya, karena ketika peneliti menguji coba tidak terjadi aktifitas apapun pada fitur ini, begitu pula yang dikatakan para pengguna lain pada *Google Play*.



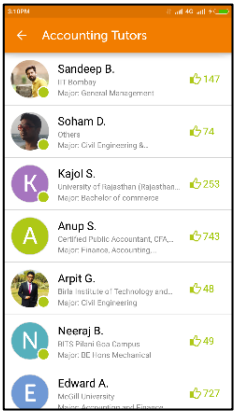
Gambar 2..7 *Screenshot* Halaman Notification

4. **Find Tutor by Subject**

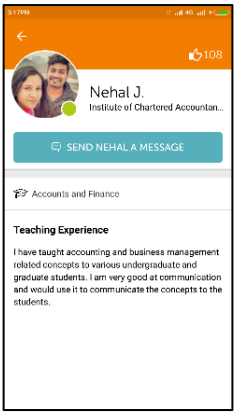
Pada fitur ini, terdapat daftar pelajaran secara umum. Pengguna dapat mencari pelajaran lain, namun yang sesuai dengan apa yang disediakan aplikasi. Setelah memilih salah satu pelajaran, maka aplikasi akan menampilkan daftar tutor yang menguasai bidang itu. Ketika telah memilih salah satu tutor kita akan diberikan informasi mengenai tutor dan dapat memulai *chat* dengan tutor tersebut.



Gambar 2.8 *Screenshot* Daftar Pelajaran



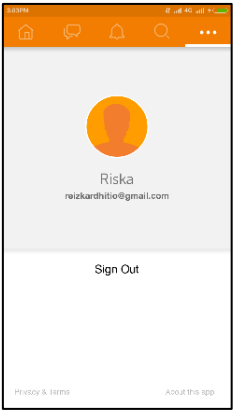
Gambar 2.9 *Screenshot* Daftar Tutor



Gambar 2.10 *Screenshot* informasi tutor

5. ***Profile***

Pada halaman ini hanya ditampilkan nama pengguna dan email pengguna. Terdapat foto *default* yang tidak bisa diganti dan juga menu *Sign Out*. Dalam fitur ***Profile*** ini tidak ada pilihan untuk mengganti nama dan email serta mengganti foto.



Gambar 2.11 *Screenshot* Halaman Profile

Berdasarkan fitur-fitur yang ada pada aplikasi *Chegg Tutor* masih memiliki kekurangan, yaitu hanya mempertemukan tutor dengan murid melalui *chat* saja, dan pelajaran yang dapat diminta hanya terbatas yang sudah disediakan oleh aplikasi *Chegg Tutor* saja.

2.2. Tutor

Tutor adalah seseorang yang memberi pelajaran atau membimbing kepada seseorang atau sejumlah kecil murid (di rumah, bukan di sekolah). Tutor sendiri memiliki makna lain yaitu dosen yang membimbing sejumlah mahasiswa dalam matakuliah [2]. Tutor atau guru les merupakan salah satu jalan bagi para orang tua dan juga para pelajar untuk meningkatkan kualitas akademik maupun non akademik.

2.3. Android

Android merupakan sistem operasi perangkat bergerak yang berbasis Linux Kernel dan saat ini sedang dikembangkan lagi oleh Google [3]. Android menggunakan antarmuka pengguna yang berbasis manipulasi langsung, dan didesain terutama untuk digunakan pada perangkat bergerak dengan layar sentuh. Android merupakan sistem operasi *opensource*, dan dirilis di bawah Lisensi Apache. Kode *opensource* dan lisensi perizinan pada Android memungkinkan perangkat lunak untuk dimodifikasi secara bebas dan didistribusikan oleh para pembuat perangkat, operator nirkabel, dan pengembang aplikasi. Selain itu, andnroid memiliki komunitas pengembangan aplikasi dengan jumlah besar yang memperluas fungsionalitas perangkat, umumnya ditulis dalam versi bahasa pemrograman java. Menurut *Google Trends* Indonesia menempati peringkat pertama dunia tentang minat dan kepopuleran android, Gambar 2.12.

Gambar 2.12 Minat Android di Dunia

Setiap tahun android selalu mengalami perubahan, baik perubahan logo, versi, fitur dan bahkan perangkat yang menggunakan sistem android. Awal mula munculnya android pada November 2007 Google merilis Android versi pertamanya dengan nama Android Alpha. Android versi 1.0 ini mulai dikomersilkan pada 22 Oktober 2008 dengan menggunakan pabrikan HTC dan diberi nama HTC Dream. Setelah munculnya android versi pertama, pihak Google mulai memberikan nama makanan ringan untuk versi selanjutnya supaya mudah diingat. Pada tahun 2009 Android memperbarui versinya hingga tiga kali, Android Cupcake, Android Donut dan juga Android Eclair. Kemudian pada tanggal 20 Mei 2010 android merilis versi barunya yaitu Android Froyo (Frozen Yoghurt). Kemudia tahun berikutnya android memperbarui empat kali versinya, Android Ginger bread, Android Honeycomb versi 3.1 dan versi 3.2, dan Android Ice Cream Sandwich. Pada versi selanjutnya Android Jelly Bean mengalami perubahan versi dari versi 4.1 hingga versi 4.3. kemudian tahun 2013 hinggan 2015 Google mengeluarkan tiga versi android, Android Kitkat, Android Lollipop dan Android Marshmallow [4]. Sampai saat ini android mengeluarkan versi terbaru yaitu Android Nougat. Daftar versi android dapat dilihat pada Gambar 2.13.

Gambar 2.13 Logo Android Berdasarkan Versi

2.4. Google Maps API

Google Maps API adalah sebuah layanan (*service*) yang diberikan oleh Google kepada para pengguna untuk memanfaatkan *Google Map* dalam mengembangkan aplikasi. *Google Maps API* menyediakan beberapa fitur, diantaranya :

1. **Google Maps API**

   Google Maps API disediakan bagi para pengguna untuk menambahkan sebuah peta yang disediakan oleh Google. Identifikasi lokasi dengan *marker*, dapat menggunakan lebih dari satu fragmen peta, dan tentunya dapat menyertakan informasi dan pemetaan yang disesuakan dengan aplikasi [5].

2. **Google Maps Geocoding API**

   Google Maps Geocoding API disediakan untuk melakukan konversi dari alamat menjadi titik koordinat geografis yang nantinya dapat digunakan untuk penempatan marker pada peta atau memosisikan peta [6].

3. **Google Maps Directions API**

Google Maps Direction API ini merupakan layanan yang disediakan Google untuk menghitung arah antar lokasi. Menggunakan transaportasi seperti angkutan umum, mengemudi, bersepeda dan berjalan juga dapat dihitung dengan menggunakan Google Maps Directions API [7].

4. **Google Place API**

Google Place API merupakan layanan yang dapat membantu untuk membuat aplikasi sesuai dengan lokasi. Terdapat beberapa *UI widget*, antara lain *UI widget PlacePicker, UI widget Autocomplete, UI widget GeoDataApi*, dan *UI widget PlaceDetectionApi*.

UI widget Autocomplete sendiri merupakan widget yang disediakan untuk membantu pengguna memperoleh hasil berupa prediksi tempat atau jalan atau kota yang akan diketikan oleh pengguna. Dalam widget ini terdapat beberapa filter yang dapat digunakan untuk memilih batasan dari prediksi yang akan diberikan [9].

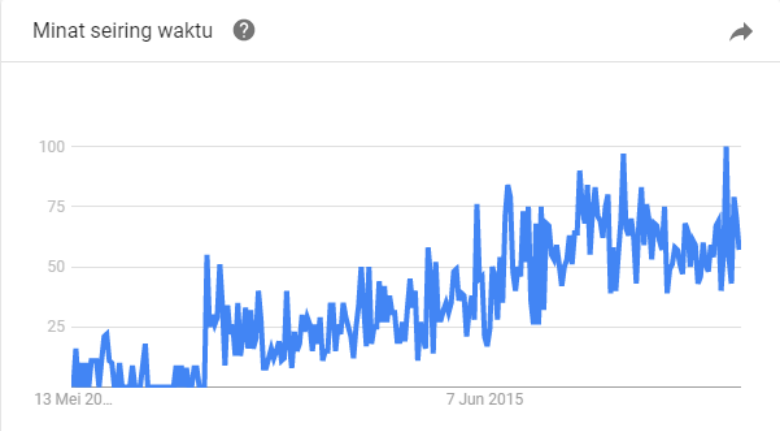
Semua fitur diatas merupakan beberapa contoh fitur yang disediakan *Google* untuk memanipulasi peta, dan menambah konten melalui berbagai jenis *services* yang dimiliki, serta mengijinkan kepada pengguna untuk membangun aplikasi *enterprise* di dalam websitenya.

Dalam penggunaannya sangatlah mudah, pengguna dapat memanfaatkan layanan-layanan yang ditawarkan oleh *Google Maps* setelah melakukan registrasi dan mendapatkan *Google Maps API Key*. Google menyediakan layanan ini secara gratis kepada pengguna di seluruh dunia .

2.5. Volley Library

Volley adalah sebuah library yang digunakan untuk membantu pertukaran data dari server dengan client. Volley

membuat pertukaran data menjadi lebih mudah dan lebih cepat [10]. Volley juga membebasakan pengguna dari penulisan kode *boilerplate* dan memungkinkan pengguna untuk berkonsentrasi pada logika yang khusus untuk aplikasi pengguna. Selain itu ketika sebuah data terdapat gambar, maka setiap gambar akan diproses dibelakang layar dan dapat mempercepat proses *request* data. Pada Gambar 2.14 dapat dilihat bahwa volley mendapati peminat yang fluktuatif [11].



Gambar 2.14 Peminat Volley Library di Dunia Lima Tahun Terakhir

Volley merupakan salah satu library yang ada untuk menjalankan *web service*. Volley sendiri memiliki beberapa keuntungan dan juga tentunya terdapat beberapa kekurangan. Keuntungan menggunakan volley sebagai *web service* adalah :

1. Volley secara otomatis menjadwalkan semua permintaan jaringan. Volley akan mengurus semua permintaan jaringan yang dijalankan aplikasi untuk mengambil respon dari web.
2. Volley menyediakan *transparent disk* dan *memory caching*.
3. Volley menyediakan API permintaan pembatalan yang kuat. Ini berarti kita dapat membatalkan permintaan tunggal atau

kita dapat menetapkan permintaan bagian mana untuk dibatalkan.

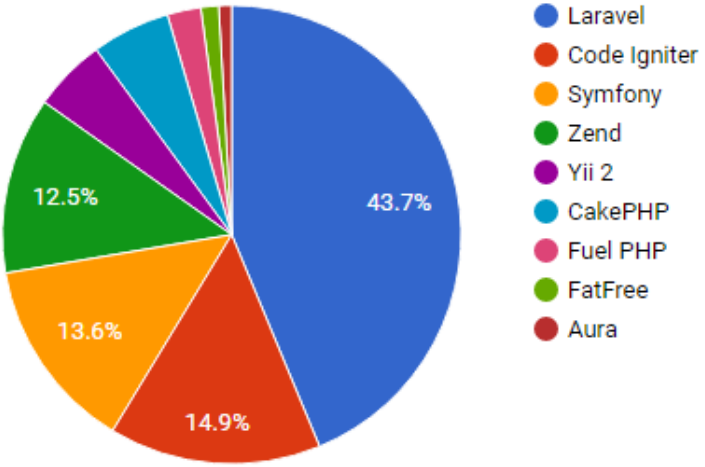
4. Volley memberikan kemampuan penyesuaian yang kuat.
5. Volley menyediakan *debugging and tracing tools* .

Namun terdapat beberapa kekurangan jika kita menggunakan Volley. Dalam hal respon, volley hanya didukung oleh *String*, *Image*, *JSONObject*, dan *JSONArray*. *Download file* ukuran besar atau *streaming* sangatlah tidak dianjurkan untuk menggunakan volley, pasalnya ketika melakukan *download file* ukuran besar atau *streaming* akan menimbulkan *out-of-memory*. Dalam hal kode, volley masih dikatakan rumit dibanding dengan *library* lain seperti retrofit.

2.6. CodeIgniter

Codeigniter adalah *web application framework* yang bersifat *open source* digunakan untuk membangun aplikasi php dinamis. Codeigniter sendiri menggunakan model MVC (Model-View-Controller) development pattern. *Framework* sendiri dapat diartikan sebagai kumpulan dari fungsi-fungsi atau prosedur-prosedur dan *class-class* yang sudah siap digunakan untuk mempermudah dan mempercepat pekerjaan programer.

Sampai saat ini, banyak sekali *framework* yang disediakan bagi para pengembang aplikasi untuk mengembangkan atau membuat aplikasi baru. Tentunya tiap *framework* sendiri memiliki kelebihan dan kekurangan. Diambil dari coderseye.com, mereka telah melakukan survei mengenai penggunaan *framework*. Hasil survei dapat dilihat pada Gambar 2.15 [13].

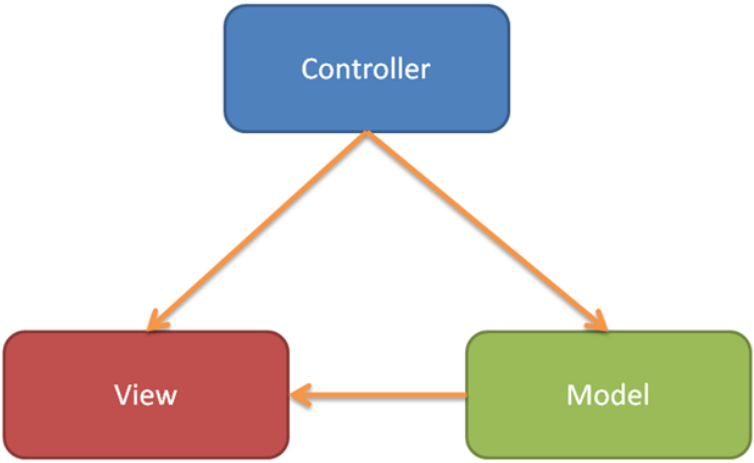


Gambar 2.15 Piechart Survei dari coderseye.com

Dapat dilihat bahwa codeigniter menempati peringkat kedua dari sembilan *framework* yang sering digunakan. Penulis memilih menggunakan codeigniter karena selain pengalaman menggunakan codeigniter, codeigniter juga memiliki beberapa kelebihan lain. Terdapat banyak *library* dan *helper* yang berguna didalamnya dan tentunya akan mempermudah proses pengembangan aplikasi. Kelebihan codeigniter yang lainnya, yaitu:

1. **URL Friendly**
   URL yang dihasilkan sangat *url friendly*. Pada codeigniter diminimalisasi penggunaan $_GET dan diganti dengan URL.
2. **Kemudahan**
   Kemudahan dalam mempelajari, membuat *library* dan *helper*, memodifikasi serta mengintegrasikan *libary* dan *helper*.

3. **Kecepatan**
   Berdasarkan hasil *benchmark* codeigniter merupakan salah satu framework PHP tercepat yang ada saat ini.
4. **Mudah dimodifikai dan beradaptasi**
   Sangat mudah untuk memodifikasi *behavior* dari codeigniter. Tidak membutuhkan *server requirement* yang macam-macam serta mudah mengadopsi *library* lainnya.
5. **Dokumentasi lengkap dan jelas**
   Codeigniter telah menyediakan dokumentasi yang sangat jelas dan lengkap. Telah disediakan informasi perbagian-bagian yang sering digunakan untuk pembuatan web atau *back-end* sebuah aplikasi.
6. **Learning curve rendah**
   Codeigniter merupakan salah satu kerangka kerja yang mudah dipelajari. Dalam pemilihan kerangka kerja sangat penting untuk memperhatikan tingkat kesulitan sebuah kerangka kerja, karena kita juga harus memperhatikan *skill* dari seluruh anggota team. Jika sebuah *framework* sangat sulit dipelajari maka akan beresiko untuk memperlambat team *development* [14].
7. **Menggunakan pattern MVC**
   MVC merupakan konsep dasar untuk penggunaan codeigniter. MVC adalah sebuah *pattern* yang memisahkan alur pikir, penyimpanan data dan juga antarmuka aplikasi atau secara sederhana adalah memisahkan antara desain, data dan proses.



Gambar 2.16 Kerangka MVC

Komponen-komponen MVC antara lain:

i. **Model**
Model berhubungan dengan data dan interaksi ke database atau webservice. Model juga merepresentasikan struktur data dari aplikasi yang bisa berupa basis data maupun data lain, misalnya file teks, file XML maupun webservice. Dalam model akan berisi class dan fungsi untuk mengambil, melakukan update dan menghapus data website.

ii. **View**
View berhubungan dengan segala sesuatu yang akan ditampilkan ke *end-user*. Bisa berupa halaman web, css, javacript dan lain-lain. Dalam view hanya berisi variable-variable yang berisi data yang siap ditampilkan. Dianjurkan untuk tidak melakukan koneksi ke basisdata. View hanya dikhususkan untuk menampilkan data-data hasil dari model dan controller.

iii. Controller

Controller bertindak sebagai penghubung data dan view. Di dalam controller terdapat class dan fungsi yang memproses permintaan dari view ke dalam struktur data di dalam model. Dalam controller juga tidak dianjurkan untuk berisi kode yang mengakses basis data. Controller bertugas untuk menyediakan berbagai variable yang akan ditampilkan pada view, memamnggil model untuk melakukan akses basis data, menyediakan penanganan kesalahan, mengerjakan proses logika dari aplikasi serta melakukan validasi atau cek terhadap input.

2.7. JSON

JSON adalah singkatan dari *Java Script Object Notation*, yaitu sebuah format untuk pertukaran data. Format ini dibuat berdasarkan bagian dari Bahasa Pemrograman Java Script, Stander ECMA-262 Edisi ke-3 -Desember 1999. JSON merupakan format teks yang tidak bergantung pada bahasa pemrograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Puthon dll. Oleh karena sifat-sifat tersebut, menjadikanJSON ideal sebagai bahasa pertukaran-data. JSON terbuat dari dua struktur:

1. Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record*), struktur (*struct*), kamus (*dictionary*), tabel hash (*hash table*), daftar berkunci (*keyed list*), atau *associative array*.
2. Daftar nilai terurutkan. Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik *array*, *vector, list*, atau *sequence*.

Pada dasarnya, semua bahasa pemrograman modern mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemrograman yang juga berdasarkan pada struktur data ini [15].

BAB III
ANALISIS DAN PERANCANGAN SISTEM

Bab ini membahas tahap analisis dan perancangan sistem yang akan dibangun. Analisis membahas semua persiapan yang akan menjadi pokok pikiran pembuatan aplikasi ini. Sedangkan perancangan sistem membahas hal-hal yang berkaitan dengan pondasi atau dasar pembuatan aplikasi, yang meliputi perancangan basis data, tampilan antar muka halaman aplikasi, hingga perancangan alur proses yang akan diimplementasikan di dalam aplikasi.

3.1. Analisis

Tahap analisis meliputi analisis masalah, analisis kebutuhan, deskripsi umum sistem, dan kasus penggunaan sistem yang dibuat.

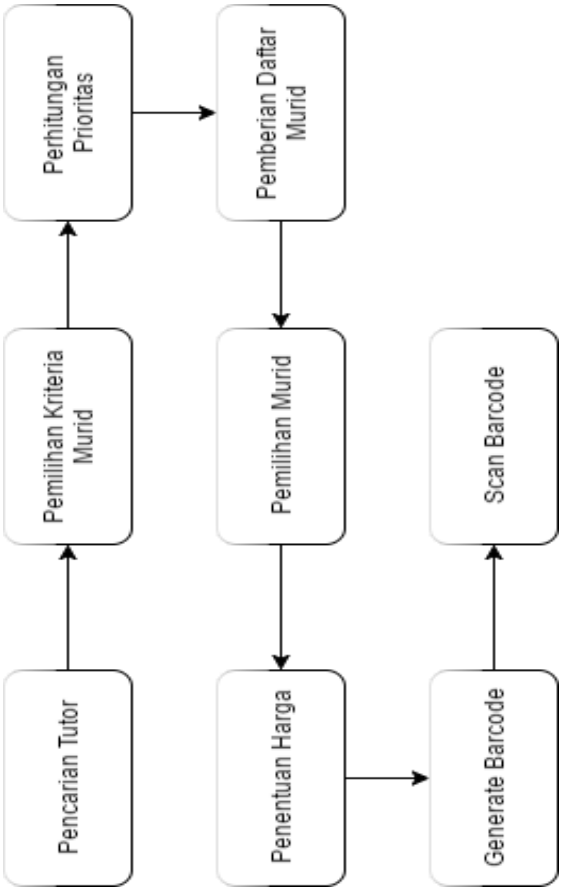
3.1.1. Deskripsi Umum Aplikasi

Aplikasi FindingTutor merupakan aplikasi android yang dibuat untuk mempermudah pencarian tutor dan pencarian murid. Murid dengan menggunakan aplikasi ini dapat memilih penyedia tutor sesuai dengan kriteria yang dimasukkan beserta dengan mata pelajaran yang ingin diajarkan. Aplikasi ini berjalan pada wilayah Surabaya. Jalannya aplikasi dapat dilihat pada Gambar 3.1, dimulai dari murid melakukan pencarian tutor dengan cara mengisi formulir pencarian tutor yang berisikan nama, alamat, mata pelajaran yang ingin diajarkan oleh tutor, alamat diadakannya tutor. Selain itu murid memasukkan kriteria tutor seperti apa yang diinginkan dengan memilih jenis kelamin tutor dan memasukkan usia tutor yang diinginkan. Kemudian tutor sendiri harus memasukkan kriteria murid terlebih dahulu untuk mendapatkan daftar prioritas murid yang didapat dari hasil perhitungan prioritas.

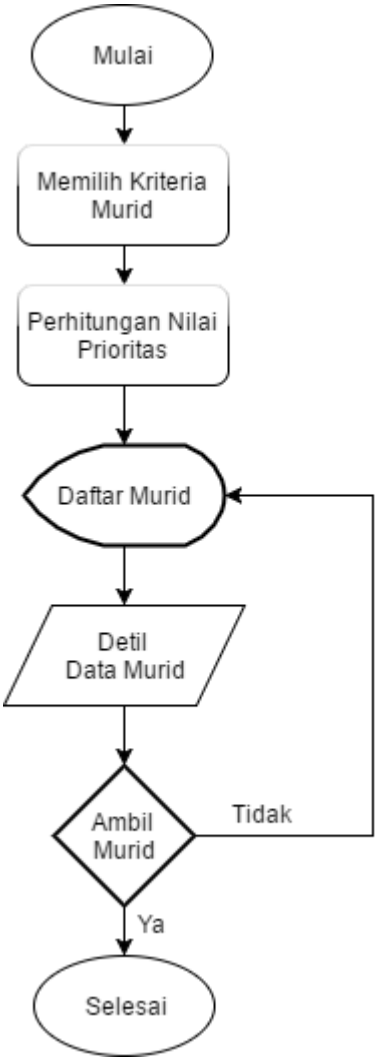
Pada aplikasi Finding-Tutor terdapat tiga modul terkait prorses pembuatan aplikasi, yaitu modul penentuan prioritas yang dibahas pada buku ini, modul perhitungan harga, dan modul

pencegahan *fraud*. Pada gambar 3.2 terdapat diagram alir dari pengguna tutor, yaitu proses pencarian murid. Untuk mendapatkan murid sesuai dengan keadaan dari tutor, pertama tutor harus memilih kriteria murid yang ingin didapatkan. Maksudnya tutor akan mendapat pilihan mulai dari jarak, mata pelajaran yang dikuasai, hingga usia tutor yang diinginkan murid. Kemudian setelah melakukan pemilihan kriteria, sistem akan melakukan perhitungan prioritas. Setelah proses perhitungan prioritas dilakukan, aplikasi akan menampilkan daftar murid. Dari daftar murid tersebut, tutor dapat memilih murid mana saja yang ingin diambil. Data murid juga akan ditampilkan setelah tutor memilih murid yang inign diambil. Data yang ditampilkan adalah data yang diisikan murid ketika melakukan pencarian tutor. Ketika tutor setuju dengan murid tersebut, maka proses pencarian murid selesai, kemudian dilanjutkan proses perhitungan harga dan penanganan *fraud*.

Kemudian pada gambar 3.3 merupakan diagram alir dari pengguna murid, yaitu proses pencarian tutor. Dalam proses pencarian tutor, pertama murid akan mengisi formulir yang akan ditampilkan aplikasi, mulai dari nama, pelajaran yang diinginkan, kelas yang diinginkan, tanggal dan jam dilakukannya tutor, alamat dilakukannya tutor, dan durasi dilakukannya tutor. Selain itu murid juga mengisikan usia dan jenis kelamin dari tutor yang diinginkan. Setelah melakukan pengisian data, murid akan mendapatkan pesan berupa *pop-up* yang berisikan estimasi biaya dari pemesanan tutor atau guru les. Setelah menyetujui estimasi biaya tersebut, data pencarian tutor akan masuk ke *database*.



Gambar 3.1 Proses Aplikasi Finding-Tutor



Gambar 3.2 Diagram Alir Pencarian Murid

Gambar 3.3 Diagram Alir Pencarian Tutor

3.1.2. Penentuan Kriteria

Sebagai seorang murid sering kali kesusahan untuk mendapatkan tutor atau guru les yang sesuai dengan keinginan murid. Finding-Tutor adalah aplikasi untuk mengatasi masalah tersebut, oleh karena itu untuk mendapatkan kriteria-kriteria yang akan digunakan pada aplikasi dilakukan penyebaran kuesioner. Penyebaran kuesioner dipilih karena dengan melakukan penyebaran kuesioner, kriteria tutor yang didapat berasal dari orang yang bersangkutan atau orang yang akan menggunakan aplikasi. Dari penyebaran kuesioner yang telah dilakukan, didapat daftar kriteria. Pada kuesioner yang telah disebar kepada masyarakat, diberikan nilai prioritas satu sampai tiga, dengan nilai satu merupakan nilai prioritas paling kecil dan nilai tiga merupakan nilai priorita tertinggi. Selanjutnya masing-masing kriteria diberikan nilai bobot yang akan menghasilkan prioritas. Kuesioner diberikan kepada mahasiswa dan mahasiswi berjumlah 18 responde serta orangtua yang berjumlah 12 responden.

Tabel 3.1 Hasil Kuesioner

No	Pertanyaan	Nilai Prioritas		
		1	2	3
1	Seberapa penting jarak tempuh dilaksanakannya tutor atau les?	3,3%	13,3%	83,3%
2	Seberapa penting jenis kelamin dari guru les atau tutor?	26,7%	36,7%	36,7%
3	Seberapa penting usia dari tutor atau guru les?	26,7%	36,7%	36,7%

Tabel 3.1 merupakan detail kriteria dan hasil dari kuesioner. Kemudian dalam kuesioner terdapat pertanyaan terbuka untuk menampung saran untuk mencari pelajar atau pencari tutor dan juga untuk mencari tutor atau guru les, serta didapat waktu

dilaksanakannya tutor atau les dan tingkat kepandaian murid. Dari hasil kuesioner tersebut, penulis menentukan daftar kriteria:

Tabel 3.2 Tabel Presentase Kriteria

Prioritas	Kriteria	Simbol	Presentase
1	Jarak	K1	75%
2	Tingkat Kepandaian Murid	K2	15%
3	Usia	K3	10%

3.1.2.1. Menghitung Bobot Persen Kriteria

Dari hasil kuesioner didapatkan urutan priritas dari masing-masing kriteria. Urutan prioritas tersebut selanjutnya akan digunakan untuk mencari bobot persen masing-masing krtieria menggunakan *Analytical Hierarchy Process* (AHP). Bobot persen akan digunakan dalam menentukan hasil prioritas.

1. Menentukan Kriteria

Tabel 3.3 Tabel Kriteria

Prioritas	Kriteria
1	K1
2	K2
3	K3

2. Membuat matriks dan memberikan nilai

Tabel 3.4 Tabel Matriks dan Nilai Kriteria

	K1	K2	K3
K1	1	3	5
K2	0,33	1	3
K3	0,2	0,33	1

3. Melakukan normalisasi

Tabel 3.5 Tabel Normalisasi Kriteria

	K1	K2	K3
K1	0,65	0,69	0,56
K2	0,22	0,23	0,33
K3	0,13	0.08	0,11

4. Mendapatkan nilai bobot

Tabel 3.6 Tabel Pemberian Bobot

	SUM	SUM/8	Bobot Persen
K1	1,9	0,63	63%
K2	0,78	0,26	26%
K3	0,32	0.11	11%

3.1.2.2. Menghitung Bobot Nilai Kriteria

Setiap Kriteria mempunyai beberapa paramater dan pada masing-masing parameter memiliki sebuah nilai. Nilai tersebut akan digunakan untuk menentukan nilai bobot dari kriteria.

Tabel 3.7 Tabel Nilai Parameter Kriteria

No	Kriteria	Bobot	Parameter	Nilai
1	Jarak	63%	≤ 3KM	100
			3KM < X ≤ 5KM	75
			5KM < X ≤ 8KM	50
			X < 8 KM	25
2	Tingkat Kepintaran Murid	26%	X = 5	100
			4 > X ≥ 3	75
			3 > X ≥ 2	50
			X < 2	25
3	Perbedaan Usia Tutor dengan Pesanan Murid	11%	Sama	100
			≤ 1 Tahun	75
			1 Tahun < X ≤ 3 Tahun	50
			X < 5 Tahun	25

Pada tugas akhir ini menggunakan rumus untuk mendapatkan nilai bobot dari murid, yaitu:

Nilai Kriteria = Nilai Parameter x Bobot Kriteria

Nilai Akhir = ∑ Nilai Kriteria

3.1.3. Analisis Kebutuhan Sistem

Kebutuhan utama dalam aplikasi ini adalah pengguna sebagai murid dapat melakukan pencarian tutor atau guru les yang sesuai dengan keinginannya, dan juga pengguna sebagai tutor atau guru les dapat mendapatkan murid yang sesuai dengan keadaannya. Secara rinci, daftar kebutuhan fungsional dapat dilihat pada table 3.8.

Tabel 3.8 Daftar Kebutuhan Fungsional Sistem

Kode Kebutuhan	Kebutuhan Fungsional	Deskripsi
F-0001	Melakukan pencarian tutor	Pengguna sebagai murid dapat melakukan pencarian tutor sesuai keinginan dari murid.
F-0002	Melihat transaksi sedang berjalan	Pengguna sebagai murid dapat melihat daftar pencarian tutor yang telah di pilih oleh tutor
F-0003	Melihat profil murid	Pengguna sebagai murid dapat melihat profil sesuai dengan yang telah diisikan.
F-0004	Mengubah profil murid	Pengguna sebagai murid dapat mengubah profil.
F-0005	Melihat *history* transaksi murid	Pengguna sebagai murid dapat melihat data *history* transaksi pemesanan tutor yang telah selesai.
F-0006	Memberikan rating dan komentar	Pengguna sebagai murid dapat memberikan rating dan komentar kepada tutor setelah semua transaksi selesai.
F-0007	Mencari murid	Pengguna sebagai tutor akan mendapatkan daftar prioritas murid sesuai dengan keadaan atau kriteria dari pentutor.
F-0008	Melihat keahlian	Pengguna sebagai tutor dapat melihat daftar keahlian dari tutor.

F-0009	Menambah keahlian	Pengguna sebagai tutor dapat menambahkan keahlian dari tutor.
F-0010	Melihat profil tutor	Pengguna sebagai tutor dapat melihat profil sesuai dengan yang telah diisikan.
F-0011	Mengubah profil tutor	Pengguna sebagai tutor dapat mengubah profil .
F-0012	Melihat *history* transaksi tutor	Pengguna sebagai tutor dapan melihat data *history* transaksi yang telah selesai beserta rating dan komentar dari murid.

3.1.4. Analisis Aktor

Aktor adalah pihak-pihak, baik manusia maupun sistem yang terlibat dan berinteraksi langsung dengan sistem. Pada aplikasi FindingTutor ini memiliki dua aktor yaitu pengguna sebagai murid dan juga pengguna sebagai tutor.

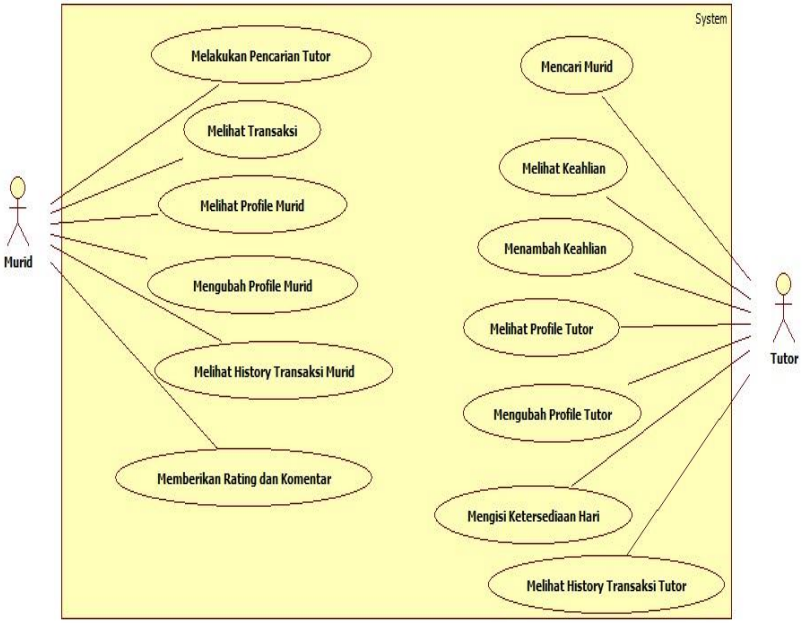
3.1.5. Kasus Penggunaan

Mengacu pada spesifikasi kebutuhan fungsional yang telah dipaparkan, dibuat kasus penggunaan yang selanjutnya akan disimpulkan dalam deskripsi umum sistem, yang diharapkan dapat memenuhi kebutuhan fungsional, berdasar pada kasus penggunaan yang dibuat. Kasus penggunaan dijelaskan lebih lanjut pada Tabel 3.9 dan diagram kasus penggunaan ditunjukkan pada Gambar 3.4.

Tabel 3.9 Daftar Kode Kasus Penggunaan

Kode Kasus Penggunaan	Nama	Aktor
UC-0001	Melakukan pencarian tutor	Murid
UC-0002	Melihat transaksi sedang berjalan	Murid
UC-0003	Melihat profil murid	Murid
UC-0004	Mengubah profil murid	Murid
UC-0005	Melihat *history* transaksi murid	Murid
UC-0006	Memberikan rating dan komentar	Murid

UC-0007	Mengisi ketersediaan hari	Tutor
UC-0008	Mencari murid	Tutor
UC-0009	Melihat keahlian	Tutor
UC-0010	Menambah keahlian	Tutor
UC-0011	Melihat profil tutor	Tutor
UC-0012	Mengubah profil tutor	Tutor
UC-0013	Melihat *history* transaksi tutor	Tutor



Gambar 3.4 Diagram Kasus Penggunaan

3.1.5.1. Melakukan Pencarian Tutor (UC-0001)

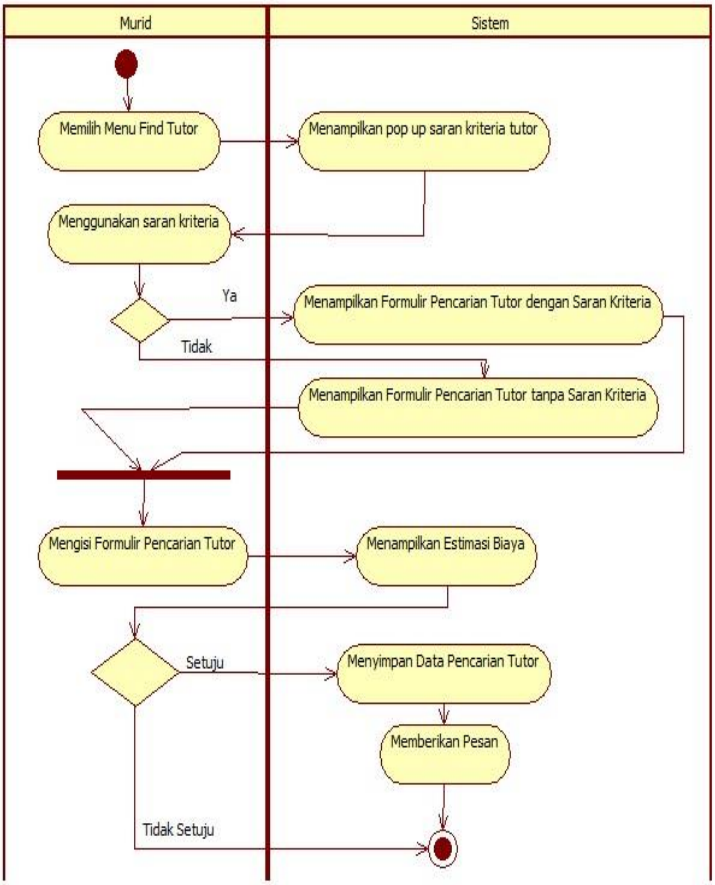
Pada kasus penggunaan ini, murid melakukan pencarian tutor. Pada pencarian tutor ini, murid dapat memilih usia dan jenis kelamin dari pentutor yang diinginkan disamping pengisian data

pencarian tutor yang lain. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.10 dan diagram aktivitas pada Gambar 3.5.

Tabel 3.10 Spesifikasi Kasus Penggunaan Melakukan Pencarian Tutor

Kode	UC-0001
Nama	Melakukan Pencarian Tutor
Deskripsi	Murid melakukan pencarian tutor
Tipe	Fungsional
Pemicu	Murid memilih menu *Find Tutor* pada aplikasi.
Aktor	Murid
Kondisi Awal	Murid telah *login*
Kondisi Akhir	Pencarian tutor telah dilakukan
Alur Kejadian Normal	- Murid memilih menu *Find Tutor* - Sistem menampilkan *pop up* kriteria otomatis - Murid menggunakan kriteria otomatis A.3. Murid tidak menggunakan kriteria otomatis - Sistem menampilkan halaman pencarian tutor dengan data kriteria tutor otomatis. - Murid mengisi formulir pencarian tutor - Sistem menampilkan estimasi biaya - Murid setuju dengan estimasi harga A.4. Murid tidak setuju dengan estimasi harga - Sistem menyimpan data pencarian tutor - Sistem memberikan pesan pencarian telah berhasil

Alur Kejadian Alternatif	A.3. Murid tidak menggunakan kriteria otomatis A.3.1. Sistem tidak memberikan kriteria tutor secara otomatis A.3.2. Kembali ke alur 5 A.4 Murid tidak setuju dengan estimasi harga A.4.1 Sistem menampilkan pesan pencarian tutor batal A.4.2 Pencarian tutor tidak disimpan.



Gambar 3.5 Diagram Aktivitas Melakukan Pencarian Tutor

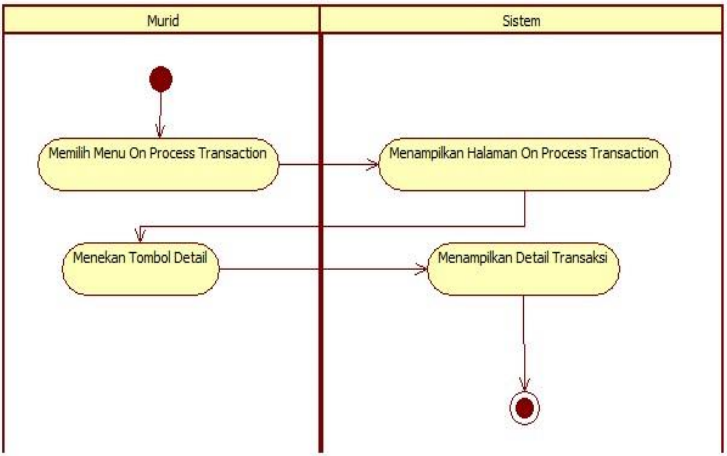
3.1.5.2. Melihat Transaksi Sedang Berjalan (UC-0002)

Pada kasus penggunaan ini, murid dapat melihat daftar transaksi yang sedang berlangsung. Murid dapat melihat data dari

pentutor yang mengambil pencarian tutor dari murid dan juga melihat harga akhir dari pencarian tutor. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.11 dan diagram aktivitas pada Gambar 3.6.

Tabel 3.11 Spesifikasi Kasus Penggunaan Melihat Transaksi Sedang Berjalan

Kode	UC-0002
Nama	Melihat Transaksi Sedang Berjalan
Deskripsi	Murid melihat daftar pencarian tutor yang sedang berlangsung
Tipe	Fungsional
Pemicu	Murid memilih menu *On Process Transaction* pada aplikasi.
Aktor	Murid
Kondisi Awal	Murid telah *login*
Kondisi Akhir	Murid mengetahui siapa pentutor yang mengambil dan harga akhir dari pencarian tutor
Alur Kejadian Normal	- Murid memilih menu *On Process Transaction* - Sistem menampilkan halaman *On Process Transaction* - Murid menekan tombol Detail - Sistem menampilkan Detail transaksi yang berisikan siapa yang mengambil dan harga akhir dari pencarian tutor
Alur Kejadian Alternatif	Tidak ada



Gambar 3.6 Diagram Aktivitas Melihat Transaksi Sedang Berjalan

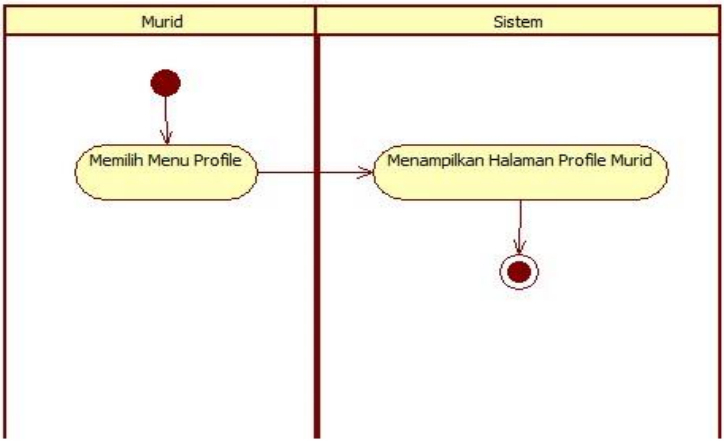
3.1.5.3. Melihat Profil Murid (UC-0003)

Pada kasus penggunaan ini, murid dapat melihat profil diri sesuai dengan data yang dimasukan ketika mendaftar akun. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.12 dan diagram aktivitas pada Gambar 3.7.

Tabel 3.12 Spesifikasi Kasus Penggunaan Melihat Profil Murid

Kode	UC-0003
Nama	Melihat Profil Murid
Deskripsi	Murid melihat profil diri
Tipe	Fungsional
Pemicu	Murid memilih menu *Profile* pada aplikasi.
Aktor	Murid
Kondisi Awal	Murid telah *login*
Kondisi Akhir	Murid mengetahui data diri yang telah diisi ketika mendaftar akun.
Alur Kejadian Normal	- Murid memilih menu *Profile*

	- Sistem menampilkan halaman *Profile*
Alur Kejadian Alternatif	Tidak ada



Gambar 3.7 Diagram Aktifitas Melihat Profil Murid

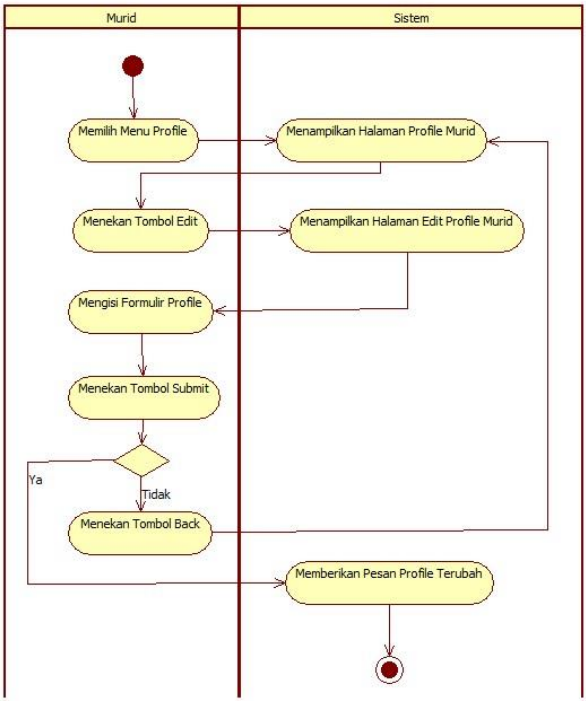
3.1.5.4. Mengubah Profil Murid (UC-0004)

Pada kasus penggunaan ini, murid dapat mengubah data diri dari murid. Data diri yang dapat diubah hanya nama, alamat, nomor telepone, dan juga email dari murid. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.13 dan diagram aktivitas pada Gambar 3.8.

Tabel 3.13 Spesifikasi Kasus Penggunaan Mengubah Profil Murid

Kode	UC-0004
Nama	Mengubah Profil Murid
Deskripsi	Murid mengubah profil diri
Tipe	Fungsional
Pemicu	Murid menekan tombol *edit* pada halaman *Profile*.
Aktor	Murid
Kondisi Awal	Murid telah berada pada halaman *Profile*

Kondisi Akhir	Murid berhasil mengubah profil.
Alur Kejadian Normal	- Murid memilih menu *Profile* - Sistem menampilkan halaman *Profile* - Murid menekan tombol edit - Sistem menampilkan halaman edit profil murid - Murid mengisi formulir profil - Murid menekan tombol submit A.6 Murid menekan tombol back - Sistem memberikan pesan profil berhasil di ubah
Alur Kejadian Alternatif	A.6 Murid menekan tombol back A.6.1 Kembali ke Alur Kejadian nomor 2.



Gambar 3.8 Diagram Aktivitas Mengubah Profil Murid

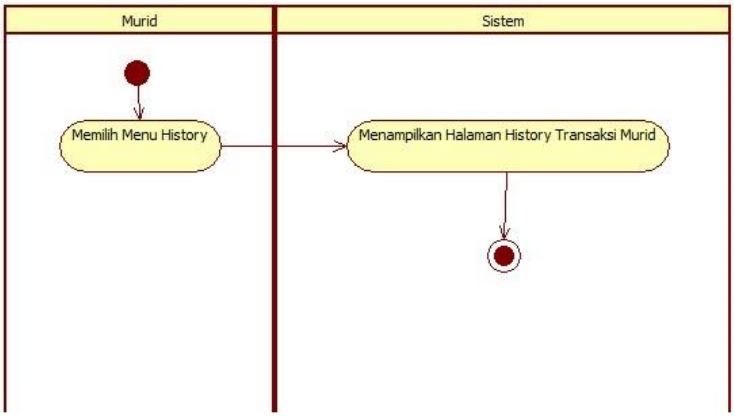
3.1.5.5. Melihat *History* Transaksi Murid (UC-0005)

Pada kasus penggunaan ini, murid dapat melihat daftar transaksi yang telah selesai. Murid dapat melihat kembali waktu melakukan tutor dan juga materi apa yang dipesan. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.14 dan diagram aktivitas pada Gambar 3.9.

Tabel 3.14 Spesifikasi Kasus Penggunaan Melihat *History* Transaksi Murid

Kode	UC-0005
Nama	Melihat *History* Transaksi Murid

Deskripsi	Murid dapat mengetahui daftar transaksi yang telah dilakukan
Tipe	Fungsional
Pemicu	Murid memilih menu *History*
Aktor	Murid
Kondisi Awal	Murid telah melakukan transaksi hingga selesai
Kondisi Akhir	Murid mengetahui daftar pemesanan tutor.
Alur Kejadian Normal	- Murid memilih menu *History* - Sistem menampilkan halaman *History*
Alur Kejadian Alternatif	Tidak ada



Gambar 3.9 Diagram Aktivitas Melihat *History* Transaksi Murid

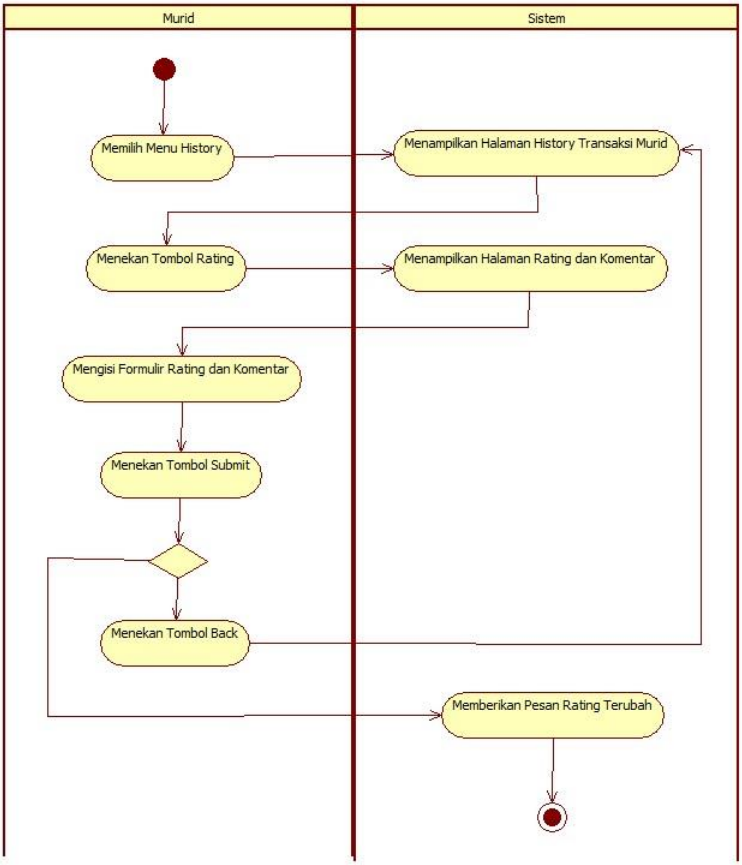
3.1.5.6. Memberikan Rating dan Komentar (UC-0006)

Pada kasus penggunaan ini, murid dapat memberikan rating dan komentar kepada pentutor. Murid akan diberikan data pencarian tutor dari nama pentutor, waktu, pelajaran dan juga biaya

dari pencarian tutor. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.15 dan diagram aktivitas pada Gambar 3.10.

Tabel 3.15 Spesifikasi Kasus Penggunaan Memberikan Rating dan Komentar

Kode	UC-0006
Nama	Memberikan rating dan komentar
Deskripsi	Murid dapat memberikan rating dan komentar kepada pentutor yang telah selesai.
Tipe	Fungsional
Pemicu	Murid menekan tombol *Rating* pada halaman *History*
Aktor	Murid
Kondisi Awal	Murid telah melakukan transaksi hingga selesai
Kondisi Akhir	Murid berhasil memberikan rating dan komentar kepada pentutor
Alur Kejadian Normal	- Murid memilih menu *History* - Sistem menampilkan halaman *History* - Murid menekan tombol *Rating* - Sistem menampilkan halaman rating dan komentar - Murid mengisi formulir rating dan komentar serta murid dapat melihat data transaksi - Murid menekan tombol *submit* A.7 Murid menekan tombol *back* - Sistem memberikan pesan bahwa rating dan komentar berhasil diberikan.
Alur Kejadian Alternatif	A.7 Murid menekan tombol *back* A.7.1 Kembali ke Alur Kejadian Normal nomor 2.



Gambar 3.10 Diagram Aktivitas Memberikan Rating dan Komentar

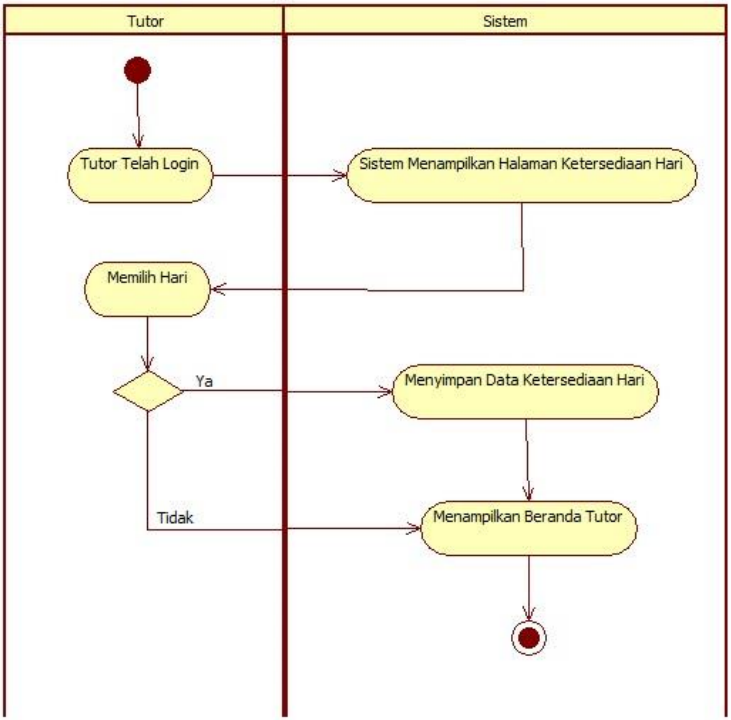
3.1.5.7. Mengisi Ketersediaan Hari (UC-0007)

Pada kasus pengguna ini, tutor dapat mengisi ketersediaan hari. Namun tutor juga dapat tidak melakukan kasus penggunaan ini bila telah mengisi ketersediaan hari sebelumnya, atau melakukan kasus pengguna edit profile. Spesifikasi kasus

penggunaan dapat dilihat pada Tabel 3.16 dan diagram aktivitas pada Gambar 3.11.

Tabel 3.16 Spesifikasi Kasus Penggunaan Mengisi Ketersediaan Hari

Kode	UC-0007
Nama	Mengisi Ketersediaan Hari
Deskripsi	Tutor dapat mengisi ketersediaan hari untuk proses pencarian murid.
Tipe	Fungsional
Pemicu	Tutor telah *login*
Aktor	Tutor
Kondisi Awal	Tutor telah *login*
Kondisi Akhir	Sistem menampilkan beranda tutor
Alur Kejadian Normal	- Tutor telah login - Sistem menampilkan halaman ketersediaan hari - Tutor memilih hari A.3 Tutor tidak memilih hari - Sistem menyimpan data ketersediaan hari - Sistem menampilkan halaman beranda
Alur Kejadian Alternatif	A.3 Tutor tidak memilih hari A.3.1 Sistem menampilkan beranda



Gambar 3.11 Diagram Aktivitas Mengisi Ketersediaan Hari

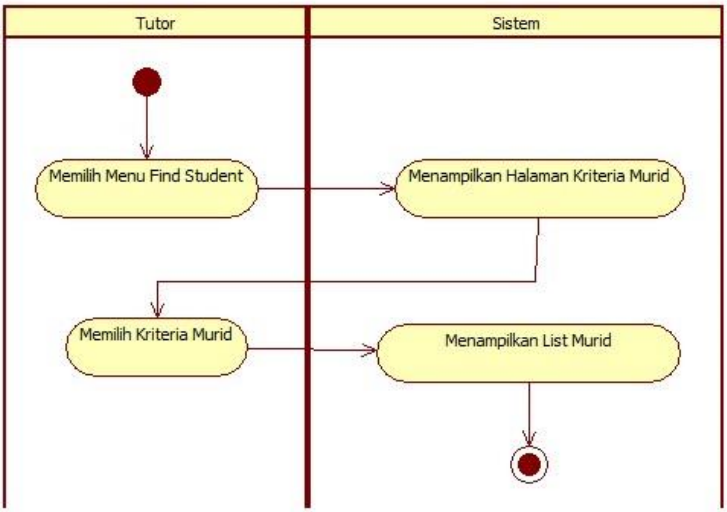
3.1.5.8. Mencari Murid (UC-0008)

Pada kasus penggunaan ini, tutor dapat mencari murid. Tutor akan mendapatkan daftar prioritas murid sesuai dengan keadaan yang dipilih. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.17 dan diagram aktivitas pada Gambar 3.12.

Tabel 3.17 Spesifikasi Kasus Penggunaan Mencari Murid

Kode	UC-0008
Nama	Mencari murid
Deskripsi	Tutor dapat mencari murid sesuai dengan keadaan dari pentutor.

Tipe	Fungsional
Pemicu	Tutor memilih menu *Find Student*
Aktor	Tutor
Kondisi Awal	Tutor telah *login*
Kondisi Akhir	Tutor mendapatkan murid sesuai keadaannya
Alur Kejadian Normal	- Tutor memilih menu *Find Student* - Sistem menampilkan halaman Kriteria Murid - Tutor menekan tombol *Submit* A.3 Tutor menekan tombol *back* - Sistem menampilkan daftar prioritas murid yang dilihat dari kriteria yang dipilih tutor
Alur Kejadian Alternatif	A.3 Tutor menekan tombol *back* A.3.1 Sistem menampilkan halaman utama



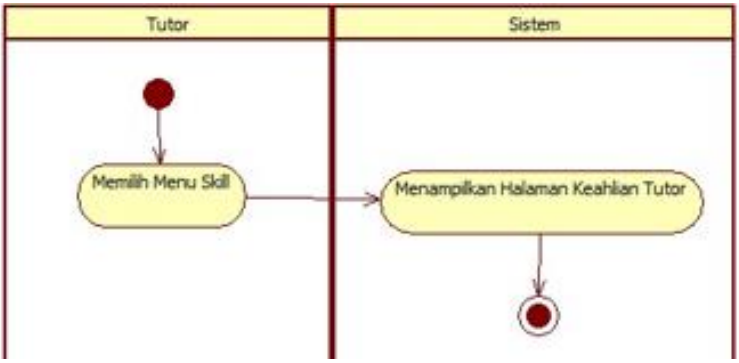
Gambar 3.12 Diagram Aktivitas Mencari Murid

3.1.5.9. Melihat Keahlian (UC-0009)

Pada kasus penggunaan ini, tutor dapat mencari murid. Tutor dapat mencari murid sesuai dengan keadaan yang dipilih. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.18 dan diagram aktivitas pada Gambar 3.13.

Tabel 3.18 Spesifikasi Kasus Penggunaan Melihat Keahlian

Kode	UC-0009
Nama	Melihat keahlian
Deskripsi	Tutor dapat melihat daftar keahlian dari tutor itu sendiri.
Tipe	Fungsional
Pemicu	Tutor memilih menu *Skill*
Aktor	Tutor
Kondisi Awal	Tutor telah *login*
Kondisi Akhir	Tutor mengetahui daftar keahlian yang telah ditambah
Alur Kejadian Normal	- Tutor memilih menu *Skill* - Sistem menampilkan halaman Keahlian Tutor
Alur Kejadian Alternatif	Tidak ada



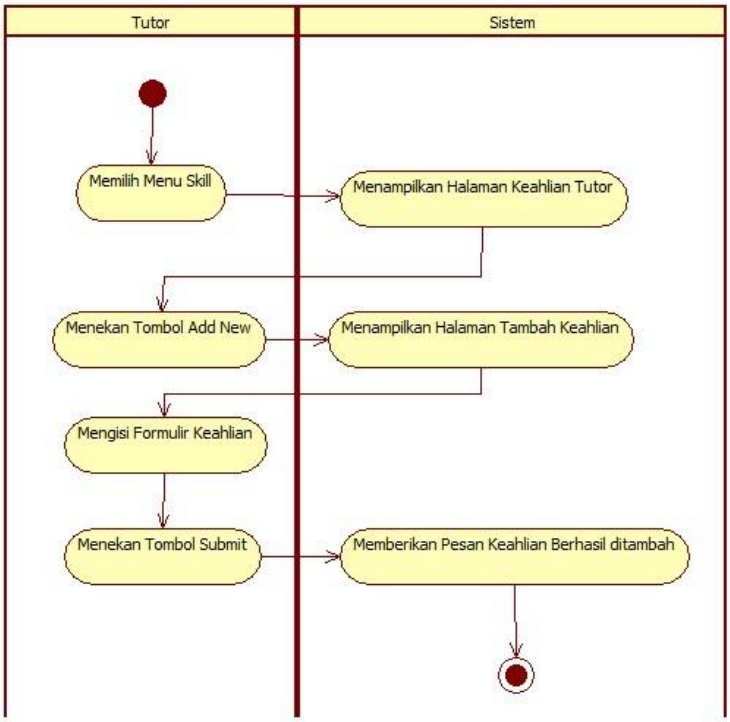
Gambar 3.13 Diagram Aktivitas Melihat Keahlian

3.1.5.10. Menambah Keahlian (UC-0010)

Pada kasus penggunaan ini, tutor dapat menambahkan keahlian. Tutor dapat memilih kelas berapa yang dia kuasai dan materi yang dia mau ambil. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.19 dan diagram aktivitas pada Gambar 3.14.

Tabel 3.19 Spesifikasi Kasus Penggunaan Menambah Keahlian

Kode	UC-0010
Nama	Menambahkan Keahlian
Deskripsi	Tutor dapat menambahkan keahlian dirinya untuk dijadikan kriteria pencarian murid
Tipe	Fungsional
Pemicu	Tutor menekan tombol *Add New* pada halaman keahlian tutor
Aktor	Tutor
Kondisi Awal	Tutor telah berada pada halaman keahlian tutor
Kondisi Akhir	Tutor berhasil menambahkan keahlian
Alur Kejadian Normal	- Tutor memilih menu *Skill* - Sistem menampilkan halaman keahlian tutor - Tutor menekan tombol *Add New* - Sistem menampilkan halaman tambah keahlian - Tutor mengisi formulir tambah keahlian - Tutor menekan tombol *Submit* A.7 Tutor menekan tombol *back* - Sistem memberikan pesan keahlian berhasil ditambah
Alur Kejadian Alternatif	A.7 Tutor menekan tombol *back* A.7.1 Keahlian tidak berhasil ditambah A.7.2 Kembali ke Alur Kejadian Normal nomor 2.



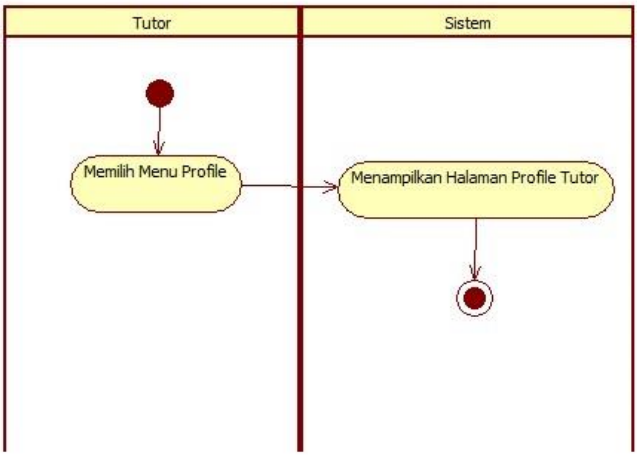
Gambar 3.14 Diagram Aktivitas Menambah Keahlian

3.1.5.11. Melihat *Profile* Tutor (UC-0011)

Pada kasus penggunaan ini, tutor dapat melihat profil diri sesuai dengan data yang dimasukan ketika mendaftar akun. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.20 dan diagram aktivitas pada Gambar 3.15.

Tabel 3.20 Spesifikasi Kasus Penggunaan Melihat Profil Tutor

Kode	UC-0011
Nama	Melihat Profil Tutor
Deskripsi	Tutor melihat profil diri
Tipe	Fungsional
Pemicu	Tutor memilih menu *Profile* pada aplikasi.
Aktor	Tutor
Kondisi Awal	Tutor telah *login*
Kondisi Akhir	Tutor mengetahui data diri yang telah diisi ketika mendaftar akun.
Alur Kejadian Normal	- Tutor memilih menu *Profile* - Sistem menampilkan halaman *Profile*
Alur Kejadian Alternatif	Tidak ada



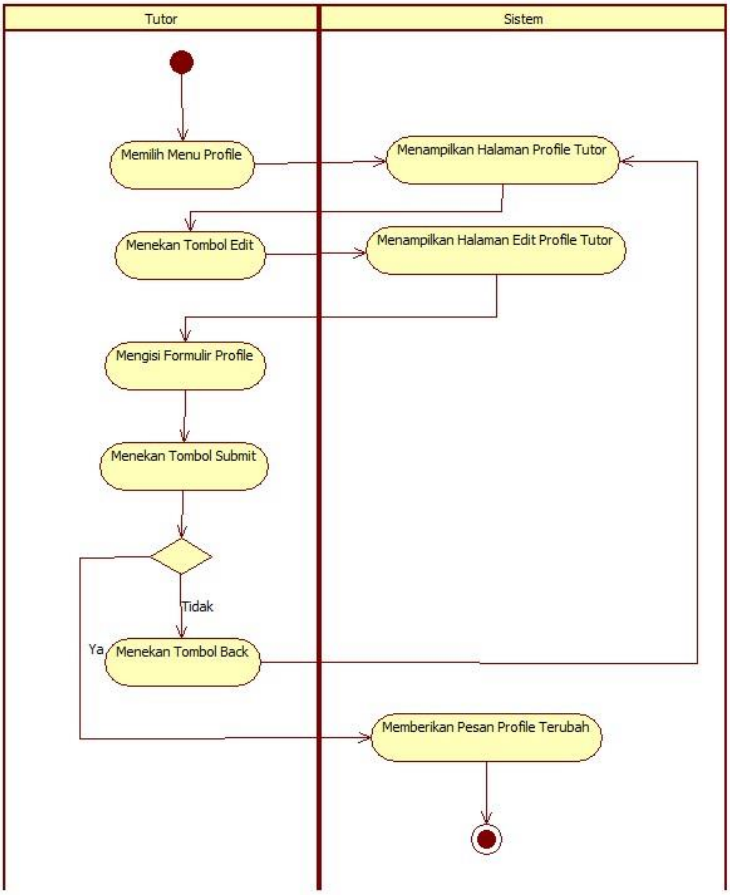
Gambar 3.15 Diagram Aktivitas Melihat *Profile* Tutor

3.1.5.12. Mengubah Profil Tutor (UC-0012)

Pada kasus penggunaan ini, tutor dapat mengubah data diri dari murid. Data diri yang dapat diubah hanya nama, alamat, nomor telepone, email dan juga ketersediaan hari dari tutor. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.21 dan diagram aktivitas pada Gambar 3.16.

Tabel 3.21 Spesifikasi Kasus Penggunaan Mengubah Profil Tutor

Kode	UC-0012
Nama	Mengubah Profil Tutor
Deskripsi	Tutor mengubah profil diri
Tipe	Fungsional
Pemicu	Tutor menekan tombol *edit* pada halaman *Profile*.
Aktor	Tutor
Kondisi Awal	Tutor telah berada pada halaman *Profile*
Kondisi Akhir	Murid berhasil mengubah profil.
Alur Kejadian Normal	- Tutor memilih menu *Profile* - Sistem menampilkan halaman *Profile* - Tutor menekan tombol edit - Sistem menampilkan halaman edit profil murid - Tutor mengisi formulir profil - Tutor menekan tombol submit A.6 Tutor menekan tombol back - Sistem memberikan pesan profil berhasil di ubah
Alur Kejadian Alternatif	A.6 Tutor menekan tombol back A.6.1 Kembali ke Alur Kejadian nomor 2.



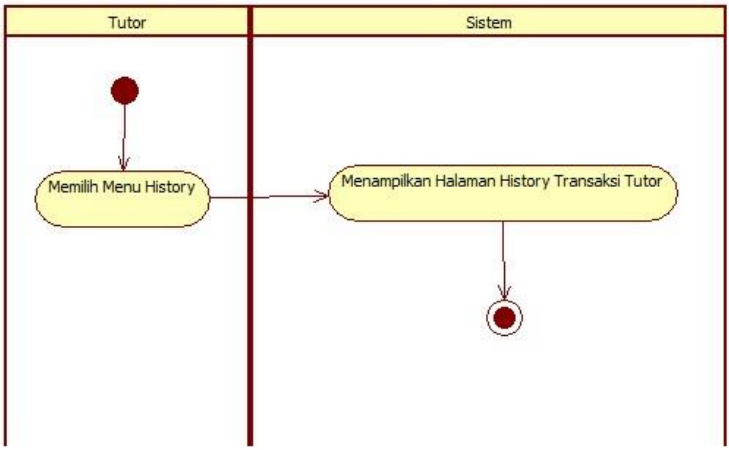
Gambar 3.16 Diagram Aktivitas Mengubah Profil Tutor

3.1.5.13. Melihat *History* Transaksi Tutor (UC-0013)

Pada kasus penggunaan ini, tutor dapat melihat daftar transaksi yang telah selesai. Tutor dapat melihat rating dan komentar dari murid yang melakukan pencarian tutor. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.22 dan diagram aktivitas pada Gambar 3.17.

Tabel 3.22 Spesifikasi Kasus Penggunaan Melihat *History* Transaksi Tutor

Kode	UC-0013
Nama	Melihat *History* Transaksi Tutor
Deskripsi	Tutor dapat mengetahui daftar transaksi yang telah dilakukan
Tipe	Fungsional
Pemicu	Tutor memilih menu *History*
Aktor	Tutor
Kondisi Awal	Tutor telah melakukan transaksi hingga selesai
Kondisi Akhir	Tutor mengetahui daftar pemesanan tutor beserta rating dan komentar.
Alur Kejadian Normal	- Tutor memilih menu *History* - Sistem menampilkan halaman *History*
Alur Kejadian Alternatif	Tidak ada



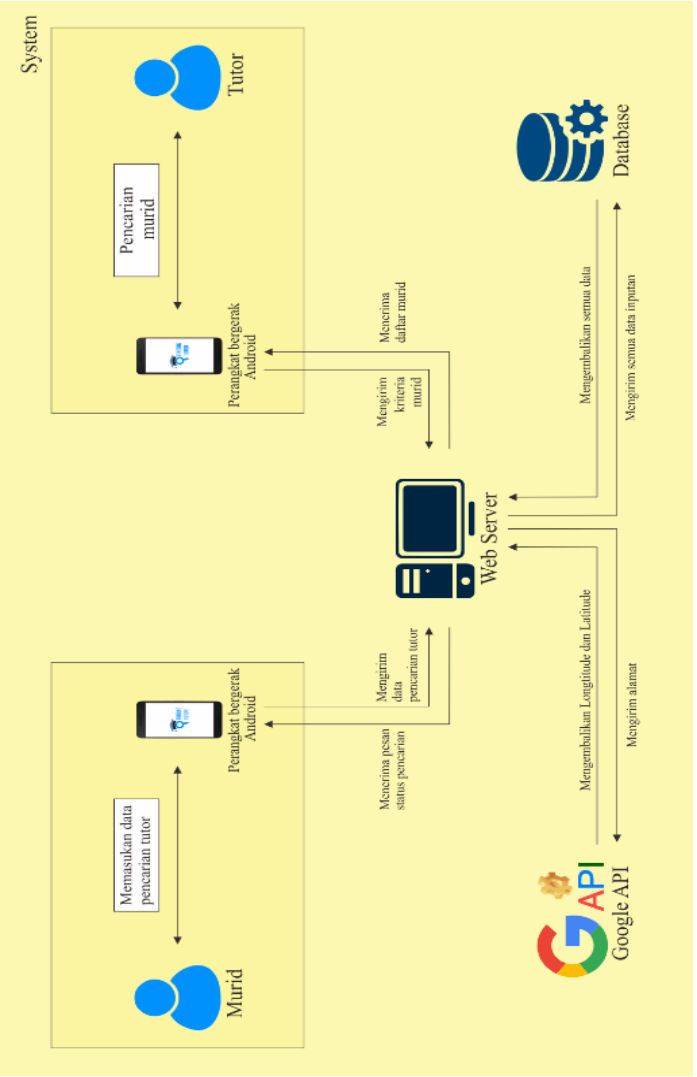
Gambar 3.17 Diagram Aktivitas Melihat *History* Transaksi Tutor

3.2. Perancangan Sistem

Tahap ini meliputi perancangan basis data, tampilan antarmuka, dan perancangan arsitektur sistem yang diharapkan dapat memenuhi tujuan dari pengembangan aplikasi ini.

3.2.1. Perancangan Arsitektur Aplikasi

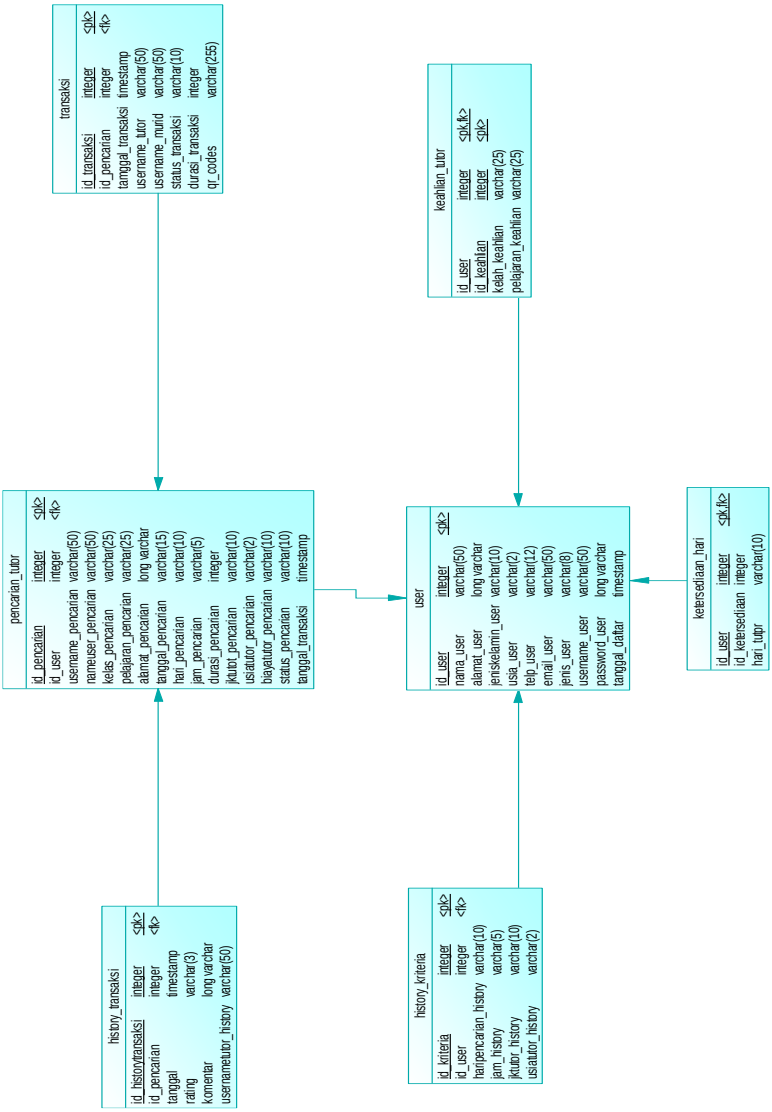
Aplikasi FindingTutor merupakan aplikasi berbasis android yang bertujuan untuk mempertemukan murid dan pentutor. Murid dapat melakukan pencarian tutor, dan tutor dapat melakukan pencarian murid. Agar data pencarian murid dapat didapatkan oleh pentutor disediakan sebuah *web server* dan *database*. *Web server* berperan untuk memninta dan mengirimkan data kepada *database*. Arsitektur sistem dapat dilihat pada Gambar 3.18.



Gambar 3.18 Perancangan Arsitektur Sistem

3.2.2. Perancangan Basis Data

Pada subbab ini dijelaskan mengenai perancangan basis data yang dalam hal ini digunakan untuk menyimpan data diri pengguna baik murid maupun tutor, proses transaksi antara murid dan tutor, keahlian tutor, *history* kriteria tutor, serta history transaksi yang dilakukan murid maupun tutor. Gambaran perancangan basis data dapat dilihat pada Gambar 3.19.



Gambar 3.19 ***Physycal Data Model***

3.2.3. Perancangan Tampilan Antarmuka

Subbab ini menjelaskan bagaimana rancangan antarmuka yang akan berinteraksi secara langsung dengan pengguna.

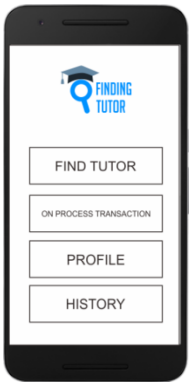
3.2.3.1. Rancangan Antarmuka Halaman Utama

Halaman ini merupakan halaman utama yang menampilkan dua menu yaitu *Sign In* dan juga *Sign Up*. Kedua menu tersebut nantinya akan disajikan dengan bentuk tombol.

Gambar 3.20 Rancangan Antarmuka Halaman Utama

3.2.3.2. Rancangan Antarmuka Halaman Beranda Murid

Halaman beranda murid ini merupakan halaman beranda bagi pengguna dengan jenis pengguna murid. Pada halaman ini terdapat empat pilihan menu dengan bentuk tombol yaitu *Find Tutor, On Process Transaction, Profile,* dan juga *History*.



Gambar 3.21 Rancangan Antarmuka Halaman *Home* Murid

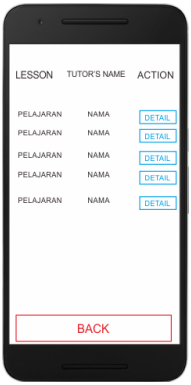
3.2.3.3. Rancangan Antarmuka Halaman *Find Tutor*

Di halaman ini sistem akan menampilkan formulir yang berisikan kelas, pelajaran, alamat, tanggal, jam, durasi, jenis kelamin tutor dan juga umur dari tutor. Terdapat tombol *Submit* yang berguna untuk memasukkan data yang telah diisi kedalam basis data dan tombol *back* yang berfungsi untuk kembali ke halaman beranda murid.

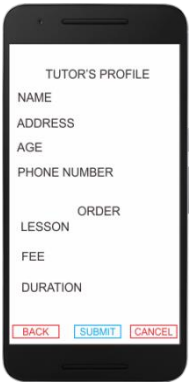
Gambar 3.22 Rancangan Antarmuka Halaman *Find Tutor*

3.2.3.4. Rancangan Antarmuka Halaman *On Process Transaction*

Pada halaman ini sistem akan menampilkan daftar dari pencarian tutor yang telah dilakukan oleh murid yang telah mendapatkan tutor. Terdapat keterangan pelajaran dan nama tutor serta tombol detail pada Gambar 3.23. Setelah murid menekan tombol detail nantinya akan menuju halaman detail transaksi pada Gambar 3.24 yang berisikan data tutor dan juga data pemesanan tutor serta harga akhir dari pemesanan tutor.



Gambar 3.23 Rancangan Antarmuka Halaman *On Process Transaction*



Gambar 3.24 Rancangan Antarmuka Halaman Detail *Transaction*

3.2.3.5. Rancangan Antarmuka Halaman *Profile* Murid

Pada halaman ini sistem akan menampilkan beberapa data yang telah dimasukkan murid ketika melakukan pendaftaran akun. Terdapat nama, alamat, nomor telepon, serta *e-mail* dari murid. Terdapat tombol *edit* yang berfungsi untuk menuju halaman *edit profile* dan tombol *back* yang berfungsi untuk kembali ke halaman beranda murid

Gambar 3.25 Rancangan Antarmuka Halaman *Profile* Murid

3.2.3.6. Rancangan Antarmuka Halaman *Edit Profile* Murid

Pada halaman *Edit Profile* sistem akan menampilkan data-data yang terdapat pada halaman *Profile* yang dapat diubah oleh murid. Terdapat tombol s*ubmit* yang dapat pengguna tekan jika merasa data yang diubah telah benar dan juga terdapat tombol *back* untuk kembali ke halaman *Profile* murid.

Gambar 3.26 Rancangan Antarmuka Halaman *Edit Profile* Murid

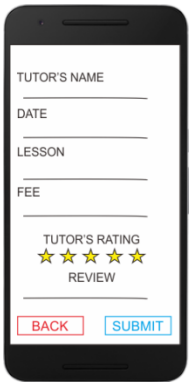
3.2.3.7. Rancangan Antarmuka Halaman *History* Murid

Pada halaman ini sistem akan menampilkan daftar *history* transaksi yang berisi waktu, dan pelajaran yang telah dipesan oleh murid. Terdapat rating bagi pentutor yang dapat diubah dengan menekan tombol detail, dan tombol *back* yang dapat ditekan untuk kembali ke halaman beranda.

Gambar 3.27 Rancangan Antarmuka Halaman *History* Murid

3.2.3.8. Rancangan Antarmuka Halaman Rating dan Komentar

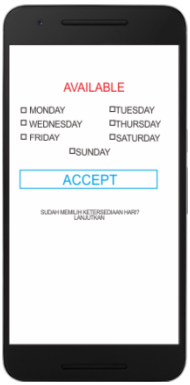
Halaman ini akan sistem tampilkan ketika murid menakan tombol detail pada halaman *History*. Halaman ini menampilkan nama tutor, tanggal dilakukannya pentutoran, pelajaran yang dipesan, dan juga biaya akhir dari pencarian tutor. Terdapat pula rating dan komentar yang dapat diubah oleh murid untuk menilai pentutor yang didapat.



Gambar 3.28 Rancangan Antarmuka Halaman Rating dan Komentar

3.2.3.9. Rancangan Antarmuka Halaman Ketersediaan Hari Tutor

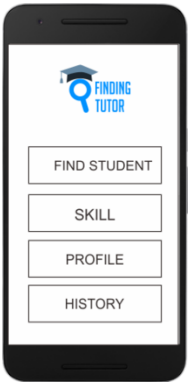
Halaman ketersediaan hari tutor ini merupakan halaman pertama yang akan ditampilkan sistem kepada pengguna tutor setelah pengguna melakukan *login*. Pada halaman ini tutor bisa tidak mengisi ketersediaan hari bila pernah mengisi sebelumnya atau mengubah pada *edit profile*.



Gambar 3.29 Rancangan Antarmuka Halaman Ketersediaan Hari Tutor

3.2.3.10. Rancangan Antarmuka Halaman Beranda Tutor

Halaman beranda tutor ini merupakan halaman beranda bagi pengguna dengan jenis pengguna tutor. Pada halaman ini terdapat empat pilihan menu dengan bentuk tombol yaitu *Find Student*, *Skill*, *Profile*, dan juga *History*

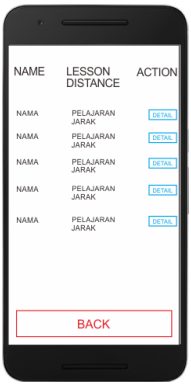


Gambar 3.30 Rancangan Antarmuka Halaman *Home* Tutor

3.2.3.11. Rancangan Antarmuka Halaman *Find Student*

Pada halaman ini sistem akan menampilkan terlebih dahulu halaman kriteria murid pada Gambar 3.31 yang terdapat pilihan berupa *dropdown* yang berguna untuk mencari murid sesuai dengan keadaan tutor. Terdapat tombol *Submit* yang berguna untuk menuju halaman pencarian murid pada Gambar 3.32 dan juga *back* untuk kembali ke beranda tutor. Pada halaman pencarian murid Gambar 3.32 terdapat daftar prioritas murid yang diambil berdasarkan keadaan yang dipilih tutor pada halaman kriteria murid. Pada halaman pencarian murid terdapat nama dari pencari tutor, pelajaran yang dipesan oleh pencari tutor, dan juga jarak antara tutor dengan pencari tutor. Terdapat tombol detail yang jika tutor tekan nantinya akan menuju halaman detail murid pada Gambar 3.33 yang berisikan data lengkap mengenai pencarian tutor mulai dari nama hingga biaya, serta terdapat peta yang menunjukan jalur bagi tutor untuk menuju ke pencari tutor.

Gambar 3.31 Rancangan Antarmuka Halaman Kriteria Murid



Gambar 3.32 Rancangan Antarmuka Halaman Daftar Murid

Gambar 3.33 Rancangan Antarmuka Halaman Detail Murid

3.2.3.12. Rancangan Antarmuka Halaman *Skill*

Pada halaman *Skill* terdapat daftar keahlian dari tutor. Data yang ditampilkan adalah pelajaran dan kelas yang dikuasai oleh tutor. Terdapat tombol *add new* yang berguna untuk menuju halaman tambah keahlian dan juga tombol *back* yang berguna untuk kembali ke beranda tutor.

Gambar 3.34 Rancangan Antarmuka Halaman *Skill*

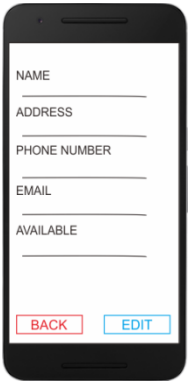
3.2.3.13. Rancangan Antarmuka Halaman Tambah Keahlian

Halaman ini akan ditampilkan sistem ketika tutor telah menekan tombol *add new* pada halaman *Skill*. Pada halaman ini terdapat *dropdown* yang berisikan kelas-kelas mulai dari kelas 1 sd hingga umum, serta kolom pelajaran berguna untuk menampung data kemampuan tutor. Terdapat tombol *submit* yang berguna untuk menambahkan data keahlian pentutor kedalam basis data, dan juga tombol *back* yang berguna untuk kembali ke halaman *Skill*.

Gambar 3.35 Rancangan Antarmuka Halaman Tambah Keahlian

3.2.3.14. Rancangan Antarmuka Halaman *Profile* Tutor

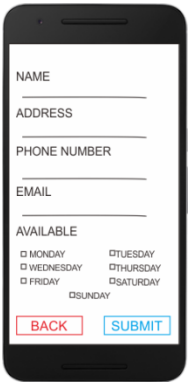
Pada halaman ini sistem akan menampilkan beberapa data yang telah dimasukkan tutor ketika melakukan pendaftaran akun. Terdapat nama, alamat, nomor telepon, serta *e-mail* dari murid. Terdapat tombol *edit* yang berfungsi untuk menuju halaman *edit profile* dan tombol *back* yang berfungsi untuk kembali ke halaman beranda tutor.



Gambar 3.36 Rancangan Antarmuka Halaman *Profile* Tutor

3.2.3.15. Rancangan Antarmuka Halaman *Edit Profile* Tutor

Pada halaman *Edit Profile* sistem akan menampilkan data-data yang terdapat pada halaman *Profile* yang dapat diubah oleh tutor. Terdapat tombol *submit* yang dapat pengguna tekan jika merasa data yang diubah telah benar dan juga terdapat tombol *back* untuk kembali ke halaman *Profile* tutor.



Gambar 3.37 Rancangan Antarmuka Halaman *Edit Profile* Tutor

3.2.3.16. Rancangan Antarmuka Halaman *History* Tutor

Halaman ini menampilkan daftar transaksi yang telah dilakukan oleh tutor. Daftar ini berisikan rating dari murid, waktu dilaksanakannya tutor, dan komentar yang diberikan oleh murid. Terdapat tombol *back* untuk kembali ke beranda

Gambar 3.38 Rancangan Antarmuka Halaman *History* Tutor

BAB IV
IMPLEMENTASI

Bab ini membahas implementasi dari analisis dan perancangan sistem yang telah dibahas pada Bab III. Namun dalam penerapannya, rancangan tersebut dapat mengalami perubahan minor sewaktu-waktu apabila dibutuhkan.

4.1. Lingkungan Implementasi

Dalam merancang aplikasi ini, digunakan beberapa perangkat pendukung yang terdiri dari perangkat keras dan perangkat lunak.

4.1.1. Lingkungan Implementasi Perangkat Keras

Terdapat dua buah perangkat keras yang digunakan dalam implementasi pengembangan aplikasi ini adalah sebagai berikut:

- Laptop

Tipe	: Asus X550J
Prosesor	: Intel® Core™ i7-CPU (3.60 GHz)
Memori Internal	: 4GB
Memori Eksternal	: 4GB

- Perangkat Bergerak

Tipe	: Redmi Note 3
Versi Android	: 5.0.2 LRX22G
Prosesor	: Octa-core (2.0 GHz)
RAM	: 3GB
Memori Internal	: 32GB

4.1.2. Lingkungan Implementasi Perangkat Lunak

Penjelasan perangkat lunak yang digunakan dalam implementasi aplikasi ini adalah sebagai berikut:

1. Microsoft Windows 10 Pro sebagai sistem operasi.
2. MySQL untuk mengimplementasikan rancangan basis data

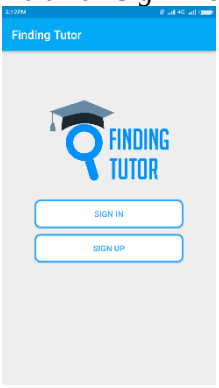
3. SublimeText sebagai kode *editor web services*
4. Android Studio sebagai kode *editor* aplikasi

4.2. Implementasi Tampilan Antarmuka

Subbab ini membahas tentang implementasi tampilan antarmuka yang telah dirancang dan dibahas pada Bab III. Selanjutnya akan dirinci berdasarkan urutan halaman yang akan tampil dan dilihat oleh partisipan.

4.2.1. Implementasi Halaman Utama

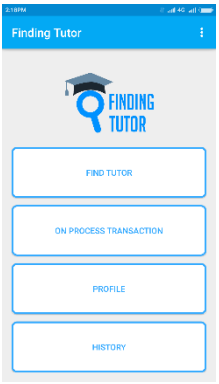
Antarmuka halaman utama merupakan halaman utama yang menampilkan dua menu yaitu *Sign In* dan juga *Sign Up*. Kedua menu tersebut disajikan dalam bentuk tombol yang jika ditekan akan menuju halaman *Sign In* dan juga *Sign Up*.



Gambar 4.1 Halaman Utama

4.2.2. Implementasi Halaman Beranda Murid

Pada halaman beranda murid ini terdapat empat menu yang berbentuk tombol yaitu, *find tutor*, *on process transaction*, *profile*, dan juga *history*.



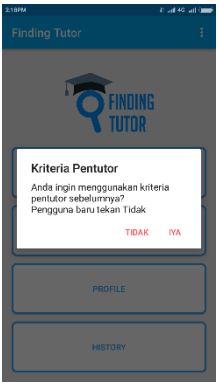
Gambar 4.2 Halaman Beranda Murid

4.2.3. Implementasi Halaman *Find Tutor*

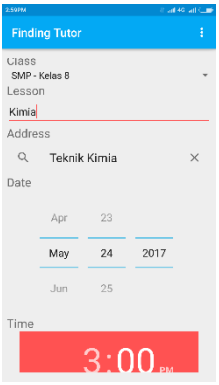
Sebelum masuk pada formulir pencarian tutor, sistem akan menampilkan *pop-up* yang berisikan pemberitahuan apakah pengguna ingin menggunakan kriteria tutor yang pernah di pesan, terdapat pada Gambar 4.3.

Jika pengguna menekan tombol iya, maka ketika sistem menampilkan formulir secara otomatis sistem akan menambahkan kriteria tutor pada kolom usia dan pilihan jenis kelamin. Jika tidak maka kolom kriteria tutor akan kosong. Selain itu terdapat pula kolom-kolom lain. Detail formulir terdapat pada Gambar 4.4 dan Gambar 4.5.

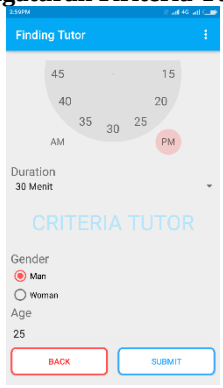
Kemudian setelah pengguna menekan tombol *submit*, sistem akan menampilkan *pop-up* yang berisikan estimasi biaya dari pencarian tutor. Jika menekan *yes*, maka pengguna dianggap setuju dan pencarian tutor akan diproses, jika menekan *no*, maka data akan dihapus dan kembali ke beranda murid. Detail estimasi pada Gambar 4.6.



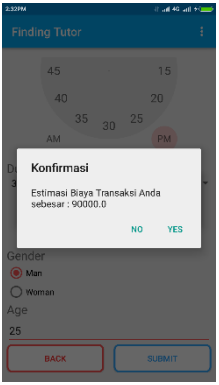
Gambar 4.3 *Pop up* Pengaturan Kriteria Tutor



Gambar 4.4 Halaman Formulir *Find Tutor*



Gambar 4.5 Halaman Formulir *Find Tutor*



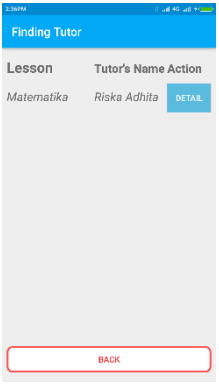
Gambar 4.6 *Pop up* Konfirmasi Pembayaran

4.2.4. Implementasi Halaman *On Process Transaction*

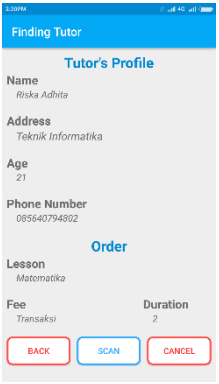
Halaman ini akan menampilkan daftar transaksi pencarian tutor yang sedang dilakukan murid, detail terdapat pada Gambar 4.7.

Kemudian jika pengguna menekan tombol detail, maka sistem akan menampilkan halaman seperti pada

Gambar 4.8 yang berisikan detail *profile* tutor, materi pembelajaran, dan juga biaya akhir pencarian tutor. Terdapat tiga tombol, tombol *back* untuk kembali ke halaman sebelumnya, tombol *scan* untuk menuju halaman *scan barcode*, dan tombol *cancel* untuk melakukan pembatalan pemesanan tutor. Detail ada pada Gambar 4.8.



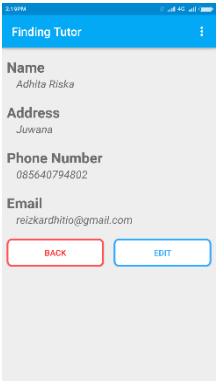
Gambar 4.7 Halaman *On Process Transaction*



Gambar 4.8 Halaman Detail *On Process Transaction*

4.2.5. Implementasi Halaman *Profile* Murid

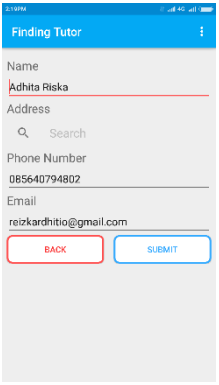
Halaman ini menampilkan *profile* dari murid. Data yang ditampilkan adalah data yang dimasukkan ketika melakukan pendaftaran (*Sign Up*). Detail halaman ada pada Gambar 4.9. Terdapat tombol *edit* untuk menuju halaman *edit profile* murid, dan tombol *back* untuk menuju halaman beranda murid.



Gambar 4.9 Halaman *Profile* Murid

4.2.6. Implementasi Halaman *Edit Profile* Murid

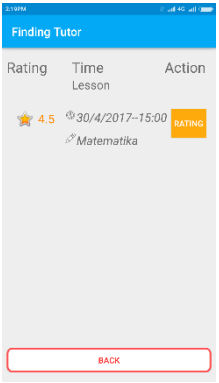
Halaman ini berisi formulir yang secara otomatis berisikan data *profile*. Pengguna dapat mengganti data-data yang ada. Setelah menekan tombol *submit* sistem akan menampilkan halaman *profile*, dengan data yang telah berubah. Pengguna dapat menekan tombol back yang akan menuju halaman *profile*, data tidak berubah. Tampilan dapat dilihat pada Gambar 4.10.



Gambar 4.10 Halaman *Edit Profile* Murid

4.2.7. Implementasi Halaman *History* Murid

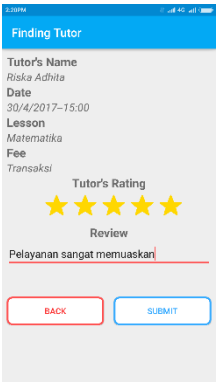
Pada halaman *history* murid, terdapat daftar *history* pencarian tutor yang telah berhasil dilakukan hingga proses akhir. Terdapat pula tombol *rating* untuk mengubah rating dan komentar bagi pentutor. Tombol *back* berguna untuk kembali ke beranda murid. Tampilan dapat dilihat pada Gambar 4.11.



Gambar 4.11 Halaman *History* Murid

4.2.8. Implementasi Halaman Rating dan Komentar

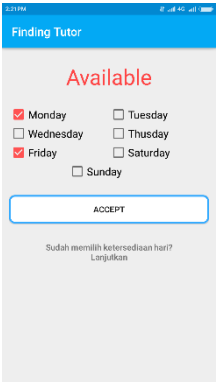
Pada halaman ini terdapat kolom komentar dan rating yang dapat diganti oleh pengguna. Terdapat informasi mengenai pencarian tutor tersebut.



Gambar 4.12 Halaman Rating dan Komentar

4.2.9. Implementasi Halaman Ketersediaan Hari Tutor

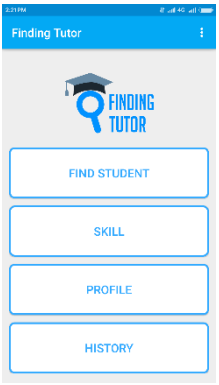
Halaman ini akan muncul pertama kali ketika pentutor *login* dan membuka aplikasi. Halaman ini berisikan pilihan hari yang akan dijadikan patokan dalam pencarian murid berdasarkan kriteria ketersediaan hari. Terdapat *link* yang dapat pengguna tekan jika tidak ingin mengubah ketersediaan hari yang pernah dimasukkan.



Gambar 4.13 Halaman Ketersediaan Hari

4.2.10. Implementasi Halaman Beranda Tutor

Terdapat empat menu pada beranda tutor yaitu, *find student*, *skill*, *profile*, dan juga *history*. Detail tampilanterdapat pada Gambar 4.14.



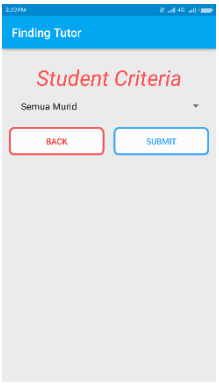
Gambar 4.14 Halaman Beranda Tutor

4.2.11. Implementasi Halaman *Find Student*

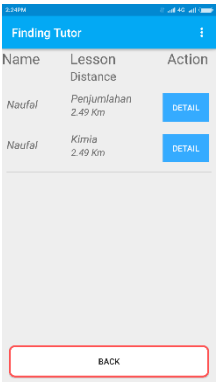
Sebelum masuk pada daftar murid yang sedang mencari tutor, sistem akan menampilkan pilihan berupa *dropdown* yang berisi kriteria pencarian murid yang ingin dicari tutor, terdapat pada Gambar 4.15.

Setelah menekan *submit*, sistem akan menampilkan daftar prioritas murid sesuai dengan kriteria yang dipilih tutor. Detail tampilan daftar murid terdapat pada Gambar 4.16.

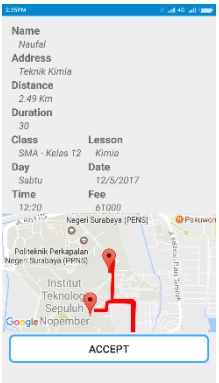
Jika pengguna menekan tombol detail yang ada pada tiap baris murid, maka sistem akan menampilkan detail informasi pencarian tutor beserta peta yang menampilkan rute dari tempat asal tutor dan alamat murid. Detail terdapat pada Gambar 4.17.



Gambar 4.15 Halaman Kriteria Pencarian Murid



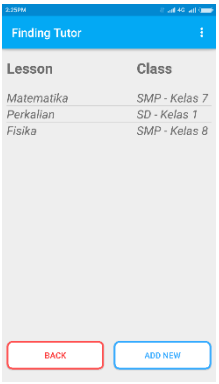
Gambar 4.16 Halaman Daftar Murid



Gambar 4.17 Halaman Detail Murid

4.2.12. Implementasi Halaman *Skill* Tutor

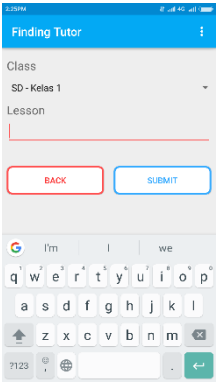
Halaman ini menampilkan daftar kompetensi dari tutor. Terdapat tombol *add* yang dapat pentutor tekan untuk menambahkan keahliannya. Detail antarmuka terdapat pada Gambar 4.18.



Gambar 4.18 Halaman *Skill* Tutor

4.2.13. Implementasi Halaman *Add Skill* Tutor

Antarmuka halaman *add skill* menampilkan kolom materi yang dapat pentutor masukkan serta pilihan kelas yang telah disediakan berupa *dropdown*. Tombol *submit* dapat pengguna tekan untuk menambahkan keahlian ke dalam basis data, dan tombol *back* dapat ditekan untuk kembali ke daftar keahlian. Detail antarmuka terdapat pada Gambar 4.19.

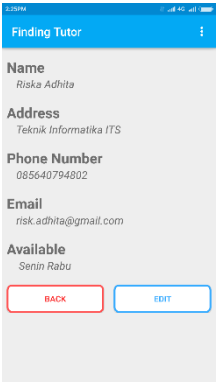


Gambar 4.19 Halaman *Add Skill* Tutor

‘

4.2.14. Implementasi Halaman *Profile* Tutor

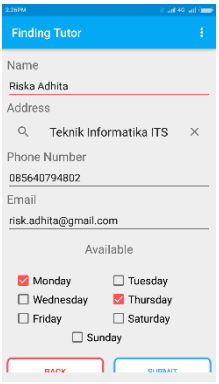
Pada halaman *profile* tutor terdapat informasi data *profile* yang tutor masukkan ketika melakukan pendaftaran (*Sign Up*). Terdapat tombol *edit* yang bila ditekan akan menuju halaman *edit profile*, dan tombol *back* yang bila ditekan akan menuju halaman beranda tutor. Detail tampilan terdapat pada Gambar 4.20.



Gambar 4.20 Halaman *Profile* Tutor

4.2.15. Implementasi Halaman *Edit Profile* Tutor

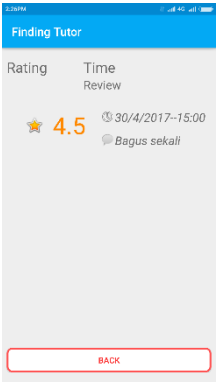
Halaman *edit profile* berisikan kolom-kolom yang secara otomatis telah berisi data *profile* dari tutor dan dapat diperbarui. Terdapat tombol *submit* yang bila ditekan akan memperbarui data *profile* dan sistem akan menampilkan halaman *profile*, dan tombol *back* yang bila ditekan akan menuju halaman *profile* tanpa merubah data *profile* tutor. Detail terdapat pada Gambar 4.21.



Gambar 4.21 Halaman *Edit Profile* Tutor

4.2.16. Implementasi Halaman *History* Tutor

Halaman *history* tutor berisikan daftar *history* transaksi yang pernah tutor lakukan hingga proses terakhir. Terdapat rating dan komentar yang diberikan oleh murid, dan terdapat waktu ketika pentutor melakukan transaksi. Terdapat tombol *back* yang bila ditekan sistem akan menampilkan beranda tutor. Detail tampilan terdapat pada Gambar 4.22.



Gambar 4.22 Halaman *History* Tutor

4.3. Implementasi Perangkat Lunak

Pada subbab ini akan dibahas mengenai implementasi alur proses aplikasi yang telah dirancang pada Bab III. Alur proses aplikasi akan dibahas mulai dari pengambilan data partisipan, hingga proses peningkatan level dan penghentian pada setiap *training*.

4.3.1. Implementasi Proses Pencarian Tutor

Pada proses pencarian tutor atau *finding tutor* ini memiliki beberapa proses. Di awali dengan proses pemberian kriteria tutor, kemudian murid dapat melakukan proses pencarian tutor dan di akhiri dengan pemberian estimasi biaya.

4.3.1.1. Proses Kriteria Tutor

Pemberian kriteria tutor di tampilkan berupa *pop up* yang berisikan apakah murid ingin menggunakan kriteria tutor yang dipesan sebelumnya atau tidak. Jika murid memilih menggunakan kriteria yang lama, maka murid akan menuju fungsi getKriteria(), dan jika tidak murid akan menggunakan fungsi deleteKriteria(). Fungsi getKriteria() dapat dilihat pada Kode Sumber 4.1, dan fungsi deleteKriteria() dapat dilihat pada Kode Sumber 4.2.

```
1   public void getKriteria()
2   {
3       StringRequest stringRequest = new
4   StringRequest(Request.Method.POST,
5                 Connect.GETKRITERIATUTOR_URL, new
6   Response.Listener<String>() {
7           @Override
8           public void onResponse(String response)
9   {
10                Log.d("coba", response);
11                try {
12                    JSONObject jsonObject = new
13  JSONObject(response);
```

```
14                  JSONArray arrayKriteria =
15  jsonObject.getJSONArray("result");
16                  JSONObject objectKriteria =
17  arrayKriteria.getJSONObject(0);
18                  jeniskelamin =
19  objectKriteria.getString("jeniskelamin");
20                  usia =
21  objectKriteria.getString("usia");
22
23                  myIntent =  new
24  Intent(getBaseContext(),CariTutorActivity.class)
25  ;
26                  Bundle bundle = new Bundle();
27                  bundle.putString("jeniskelamin",
28  jeniskelamin);
29                  bundle.putString("usia", usia);
30
31  myIntent.putExtra("bundle",bundle);
32
33  startActivityForResult(myIntent,0);
34
35              } catch (JSONException e) {
36                  e.printStackTrace();
37              }
38          }
39      },
40              new Response.ErrorListener() {
41                  @Override
42                  public void
43  onErrorResponse(VolleyError error) {
44
45  Toast.makeText(getApplicationContext(),error.get
46  Message(),Toast.LENGTH_LONG).show();
47                  }
48              }){
49          @Override
50          protected Map<String, String>
51  getParams() throws AuthFailureError {
52              Map<String, String> params = new
53  HashMap<>();
54
55  params.put("id_user",db.getIduser());
56              return params;
57          }
```

```
58      };
59      RequestQueue requestQueue =
60  Volley.newRequestQueue(this);
61      requestQueue.add(stringRequest);
62  }
```

Kode Sumber 4.1 Fungsi getKriteria()

Pada fungsi getKriteria(), *volley library* akan mengirimkan *id_user* dari murid dan akan mengambil data kriteria tutor yang pernah dipesan sebelumnya pada basis data melalui *web service*. Kemudian *web service* akan mengirimkan *respon* berupa *JSON* yang berisi jenis kelamin dan usia dari kriteria tutor yang diinginkan. Selanjutnya data yang diambil akan ditampilkan pada halaman pencarian tutor.

```
1   public void deleteKriteria()
2   {
3       StringRequest stringRequest = new
4   StringRequest(Request.Method.POST,
5   Connect.DELETEKRITERIA_URL,
6               new Response.Listener<String>() {
7                   @Override
8                   public void onResponse(String
9   response) {
10                      try {
11                          JSONObject jsonObject =
12  new JSONObject(response);
13
14  Toast.makeText(getApplicationContext(),jsonObjec
15  t.getString("message"),Toast.LENGTH_LONG).show()
16  ;
17                      } catch (JSONException e) {
18                          e.printStackTrace();
19                      }
20                  }
21              },
22              new Response.ErrorListener() {
```

```
23                  @Override
24                  public void
25  onErrorResponse(VolleyError error) {
26
27  Toast.makeText(getApplicationContext(),error.get
28  Message(),Toast.LENGTH_LONG).show();
29                  }
30              }){
31          @Override
32          protected Map<String, String>
33  getParams() throws AuthFailureError {
34              Map<String, String> params = new
35  HashMap<>();
36
37  params.put("id_user",db.getIduser());
38              return params;
39          }
40      };
41      RequestQueue requestQueue =
42  Volley.newRequestQueue(this);
43      requestQueue.add(stringRequest);
44      myIntent =  new
45  Intent(getBaseContext(),CariTutorActivity.class)
46  ;
47      Bundle bundle = new Bundle();
48      bundle.putString("jeniskelamin",
49  jeniskelamin);
50      bundle.putString("usia", usia);
51      myIntent.putExtra("bundle",bundle);
52      startActivityForResult(myIntent,0);
53  }
```

Kode Sumber 4.2 Fungsi deleteKriteria()

Pada fungsi deleteKriteria(), *volley library* juga mengirimkan *id_user* murid melalui *web service* untuk menghapus kriteria tutor yang pernah dipesan. Kemudian *web service* mengirimkan *respon* berupa *JSON* yang berisikan pesan bahwa kriteria telah dihapus dan akan diperbarui ketika melakukan pemesanan kembali.

4.3.1.2. Proses Pencarian Tutor

Pada proses pencarian tutor, murid akan memasukkan semua data yang diperlukan, mulai dari pelajaran hingga usia dari tutor. Kemudian pada fungsi cariTutor() pertama akan diambil data yang telah diisikan oleh murid dan dengan menggunakan *volley library* akan mengirimkan data tersebut untuk disimpan pada basis data melalui *web service*. Sebelum data disimpan pada basis data, sistem memberikan *pop up* yang berisikan estimasi biaya. Jika murid setuju maka akan dilakukan penyimpanan data, jika tidak maka sistem akan mengembalikan murid ke halaman beranda murid. Fungsi cariTutor() dan estimasi biaya dapat dilihat pada Kode Sumber 4.3.

```
1   private void cariTutor()
2   {
3
4       Log.d("jenis kelamin",kriteriaJenis);
5       int selectedId =
6   jeniskelamin.getCheckedRadioButtonId();
7       jkTutor = (RadioButton)
8   findViewById(selectedId);
9       final String getUsername, getNameuser, jam,
10  menit;
11      getUsername = db.getUsername();
12      getNameuser = db.getNameuser();
13      final String getTanggal;
14      getKelas = pilihKelas;
15      getPelajaran =
16  pelajaran.getText().toString();
17      month = month+1;
18      getTanggal = day+"/"+month+"/"+year;
19      toGetDay = selectedDay.toString();
20      jam =
21  String.format("%02d",waktu.getCurrentHour());
22      menit =
23  String.format("%02d",waktu.getCurrentMinute());
24      getWaktu = jam+":"+menit;
25      getJeniskelamin =
26  jkTutor.getText().toString();
27      getUsia = usia.getText().toString();
```

```
28      getDurasi = pilihDurasi;
29
30      if (toGetDay.equals("1"))
31      {
32          getHari = "Minggu";
33      }
34      else if(toGetDay.equals("2"))
35      {
36          getHari = "Senin";
37      }
38      else if(toGetDay.equals("3"))
39      {
40          getHari = "Selasa";
41      }
42      else if(toGetDay.equals("4"))
43      {
44          getHari = "Rabu";
45      }
46      else if(toGetDay.equals("5"))
47      {
48          getHari = "Kamis";
49      }
50      else if(toGetDay.equals("6"))
51      {
52          getHari = "Jumat";
53      }
54      else if(toGetDay.equals("7"))
55      {
56          getHari = "Sabtu";
57      }
58
59      progressDialog.setMessage("Pencarian
60  Tutor...");
61      progressDialog.show();
62      TingkatKesulitan tingkatKesulitan = new
63  TingkatKesulitan();
64      int kesulitan = 0;
65      double hargaawal = 0;
66
67      if (getKelas.equals("SD - Kelas 1"))
68      {
69          kesulitan = 1;
70      }
71      else if (getKelas.equals("SD - Kelas 2"))
```

```
72          {
73              kesulitan = 2;
74          }
75          else if (getKelas.equals("SD - Kelas 3"))
76          {
77              kesulitan = 3;
78          }
79          else if (getKelas.equals("SD - Kelas 4"))
80          {
81              kesulitan = 4;
82          }
83          else if (getKelas.equals("SD - Kelas 5"))
84          {
85              kesulitan = 5;
86          }
87          else if (getKelas.equals("SD - Kelas 6"))
88          {
89              kesulitan = 6;
90          }
91          else if (getKelas.equals("SMP - Kelas 7"))
92          {
93              kesulitan = 7;
94          }
95          else if (getKelas.equals("SMP - Kelas 8"))
96          {
97              kesulitan = 8;
98          }
99          else if (getKelas.equals("SMP - Kelas 9"))
100         {
101             kesulitan = 9;
102         }
103         else if (getKelas.equals("SMA - Kelas 10"))
104         {
105             kesulitan = 10;
106         }
107         else if (getKelas.equals("SMA - Kelas 11"))
108         {
109             kesulitan = 11;
110         }
111         else if (getKelas.equals("SMA - Kelas 12"))
112         {
113             kesulitan = 12;
114         }
115         else if (getKelas.equals("UMUM"))
```

```
116     {
117         kesulitan = 13;
118     }
119
120     AlertDialog.Builder alertDialogBuilder = new
121 AlertDialog.Builder(context);
122
123     alertDialogBuilder.setTitle("Konfirmasi");
124
125     // set dialog message
126     alertDialogBuilder
127             .setMessage("Estimasi Biaya
128 Transaksi Anda sebesar : " + hargafix)
129             .setCancelable(false)
130             .setPositiveButton("Yes",new
131 DialogInterface.OnClickListener() {
132                 public void
133 onClick(DialogInterface dialog,int id) {
134                     StringRequest stringRequest
135 = new StringRequest(Request.Method.POST,
136 Connect.PENCARIANTUTOR_URL,
137                             new
138 Response.Listener<String>() {
139                                 @Override
140                                 public void
141 onResponse(String response) {
142
143 progressDialog.dismiss();
144                                     try {
145
146 Log.d("respon :", response.toString());
147
148 JSONObject jsonObject = new
149 JSONObject(response);
150                                         if
151 (db.selectFlag().equals("punish"))
152
153 db.updateFlag(null);
154 Toast.makeText(getApplicationContext(),jsonObjec
155 t.getString("message"),Toast.LENGTH_LONG).show()
156 ;
157
158 finish();
159
```

```
160                                     } catch
161 (JSONException e) {
162
163 e.printStackTrace();
164                                     }
165
166                                 }
167                             },
168                             new
169 Response.ErrorListener() {
170                                 @Override
171                                 public void
172 onErrorResponse(VolleyError error) {
173
174 progressDialog.hide();
175
176 Toast.makeText(getApplicationContext(),error.get
177 Message(),Toast.LENGTH_LONG).show();
178
179 Log.d("cek",error.getMessage());
180                                 }
181                             }){
182                         @Override
183                         protected Map<String,
184 String> getParams() throws AuthFailureError {
185                             Map<String, String>
186 params = new HashMap<>();
187 params.put("id_user",db.getIduser());
188 params.put("name",getNameuser);
189 params.put("kelas",getKelas);
190 params.put("pelajaran",getPelajaran);
191 params.put("alamat",getAlamat);
192 params.put("tanggal", getTanggal);
193 params.put("hari",getHari);
194 params.put("jam",getWaktu);
195 params.put("durasi",getDurasi);
196 params.put("jeniskelamin",getJeniskelamin);
197 params.put("usia",getUsia);
198 params.put("biaya", flag);
199                             return params;
200                         }
201                     };
202                     RequestQueue requestQueue =
203 Volley.newRequestQueue(CariTutorActivity.this);
```

```
204
205 requestQueue.add(stringRequest);
206                 }
207             })
208             .setNegativeButton("No",new
209 DialogInterface.OnClickListener() {
210                 public void
211 onClick(DialogInterface dialog,int id) {
212                     dialog.cancel();
213
214 Toast.makeText(CariTutorActivity.this,
215 "Pencarian Tutor Batal",
216
217 Toast.LENGTH_LONG).show();
218                     finish();
219                 }
220             });
221
222     AlertDialog alertDialog =
223 alertDialogBuilder.create();
224     alertDialog.show();
225 }
```

Kode Sumber 4.3 Proses Pencarian Tutor

4.3.2. Implementasi Proses Melihat Transaksi Sedang Berjalan

Secara fungsional proses ini digunakan pengguna sebagai murid untuk melihat semua transaksi yang telah pengguna pesan dan diambil oleh pentutor. Pada fungsi getTransaksi() sistem mengirimkan id_user dari murid yang nantinya akan dipakai untuk menampilkan daftar transaksi yang sedang berjalan berdasarkan id_user tersebut. Kode program dapat dilihat pada Kode Sumber 4.4.

```
1   private void getTransaksi(final String id_user)
2   {
3       listTransaksiMuridDatas = new ArrayList<>();
4       StringRequest stringRequest = new
5   StringRequest(Request.Method.POST,
6               Connect.LISTTRANSAKSIMURID, new
7   Response.Listener<String>() {
8           @Override
9           public void onResponse(String response)
10  {
11              Log.d("coba", response);
12              try {
13                  JSONObject jsonObject = new
14  JSONObject(response);
15                  if(jsonObject.length() > 0)
16                  {
17                      JSONObject arrayTransaksi =
18  jsonObject.getJSONObject("result");
19                      Log.d("result",
20  arrayTransaksi.toString());
21                      String message =
22  arrayTransaksi.getString("message");
23                      Log.d("lala", message);
24                      if (message.equals("Tidak
25  ada transaksi sedang berjalan"))
26                      {
27
28  Toast.makeText(getApplicationContext(),message,T
29  oast.LENGTH_LONG).show();
30                      }
31                      JSONArray arrayListTransaksi
32  = arrayTransaksi.getJSONArray("list");
33
34                      for (int i=0; i<
35  arrayListTransaksi.length();i++)
36                      {
37                          JSONObject
38  objectTransaksi =
39  arrayListTransaksi.getJSONObject(i);
40                          ListTransaksiMuridData
41  dataTransaksi = new ListTransaksiMuridData(
42
43  objectTransaksi.getInt("id_transaksi"),
44
```

```
45  objectTransaksi.getInt("id_pencariantutor"),
46
47  objectTransaksi.getString("pelajaran"),
48
49  objectTransaksi.getString("nama_tutor"));
50
51  listTransaksiMuridDatas.add(dataTransaksi);
52                      }
53                      mAdapter = new
54  ListTransaksiMuridAdapter(ListTransaksiMuridActi
55  vity.this, 0, listTransaksiMuridDatas);
56
57  listView.setAdapter(mAdapter);
58                  }
59                  mAdapter.notifyDataSetChanged();
60
61              } catch (JSONException e) {
62                  e.printStackTrace();
63              }
64          }
65      },
66              new Response.ErrorListener() {
67                  @Override
68                  public void
69  onErrorResponse(VolleyError error) {
70
71  Toast.makeText(getApplicationContext(),error.get
72  Message(),Toast.LENGTH_LONG).show();
73                  }
74              }){
75          @Override
76          protected Map<String, String>
77  getParams() throws AuthFailureError {
78              Map<String, String> params = new
79  HashMap<>();
80              params.put("id_user",id_user);
81              return params;
82          }
83      };
84      RequestQueue requestQueue =
85  Volley.newRequestQueue(this);
86      requestQueue.add(stringRequest);
87  }
```

Kode Sumber 4.4 Proses Melihat Transaksi sedang Berjalan

4.3.3. Implementasi Proses Melihat *Profile* Murid

Proses ini terjadi bagi pengguna murid untuk melihat data diri mereka. Pada fungsi getProfileMurid() sistem mengirimkan id_user yang berguna untuk mengambil data dari basis data sesuai dengan id_user tersebut. Kode program dapat dilihat pada kode sumber 4.5.

```
1   public void getProfileMurid(final String
2   id_user)
3   {
4       StringRequest stringRequest = new
5   StringRequest(Request.Method.POST,
6               Connect.PROFILE_URL, new
7   Response.Listener<String>() {
8           @Override
9           public void onResponse(String response)
10  {
11                  try {
12                      JSONObject jsonObject = new
13  JSONObject(response);
14                      JSONArray arrayKeahlian =
15  jsonObject.getJSONArray("result");
16                      JSONObject objectProfile =
17  arrayKeahlian.getJSONObject(0);
18                      nama =
19  objectProfile.getString("nama");
20                      alamat =
21  objectProfile.getString("alamat");
22                      notelp=
23  objectProfile.getString("telp");
24                      email=
25  objectProfile.getString("email");
26
27  setView(nama,alamat,notelp,email);
28
29                  } catch (JSONException e) {
30                      e.printStackTrace();
31                  }
32          }
33      },
34                  new Response.ErrorListener() {
```

```
35                  @Override
36                  public void
37  onErrorResponse(VolleyError error) {
38
39  Toast.makeText(getApplicationContext(),error.get
40  Message(),Toast.LENGTH_LONG).show();
41                  }
42              }){
43          @Override
44          protected Map<String, String>
45  getParams() throws AuthFailureError {
46              Map<String, String> params = new
47  HashMap<>();
48              params.put("id_user",id_user);
49              params.put("jenis",jenis);
50              return params;
51          }
52      };
53      RequestQueue requestQueue =
54  Volley.newRequestQueue(this);
55      requestQueue.add(stringRequest);
56  }
```

Kode Sumber 4.5 Proses Melihat *Profile* Murid

4.3.4. Implementasi Proses Mengubah *Profile* Murid

Pada proses ini, murid akan mengisi formulir yang sudah tersedia. Pada formulir ini sistem menampilkan data yang belum terganti supaya murid hanya mengganti data diri yang ingin diganti saja. Kemudian setelah mengisi formulir, maka *volley library* akan mengirimkan data ke basis data melalui *web service*. Kode program dapat dilihat pada kode sumber 4.6.

```
1   public void updateUser()
2   {
3
4       nama = etnama.getText().toString();
5       email = etemail.getText().toString();
6       notelp = etnotelp.getText().toString();
7
```

```
8       progressDialog.setMessage("Ubah Profil...");
9       progressDialog.show();
10      StringRequest stringRequest = new
11  StringRequest(Request.Method.POST,
12  Connect.EDITPROFILE_URL,
13              new Response.Listener<String>() {
14                  @Override
15                  public void onResponse(String
16  response) {
17                      progressDialog.dismiss();
18                      try {
19                          JSONObject jsonObject =
20  new JSONObject(response);
21
22  Toast.makeText(getApplicationContext(),jsonObjec
23  t.getString("message"),Toast.LENGTH_LONG).show()
24  ;
25
26  db.updateAlamat(username,alamat);
27                          finish();
28                      } catch (JSONException e) {
29                          e.printStackTrace();
30                      }
31                  }
32              },
33              new Response.ErrorListener() {
34                  @Override
35                  public void
36  onErrorResponse(VolleyError error) {
37                      progressDialog.hide();
38                      error.printStackTrace();
39
40  Toast.makeText(getApplicationContext(),error.get
41  Message(),Toast.LENGTH_LONG).show();
42                  }
43              }){
44          @Override
45          protected Map<String, String>
46  getParams() throws AuthFailureError {
47              Map<String, String> params = new
48  HashMap<>();
49
50  params.put("id_user",db.getIduser());
51              params.put("nama",nama);
```

```
52                params.put("alamat",alamat);
53                params.put("telp",notelp);
54                params.put("email",email);
55
56    params.put("ketersediaanhari","NULL");
57                return params;
58            }
59        };
60        RequestQueue requestQueue =
61    Volley.newRequestQueue(this);
62        requestQueue.add(stringRequest);
63    }
```

Kode Sumber 4.6 Proses Mengubah *Profile* Murid

4.3.5. Implementasi Proses Melihat *History* Transaksi Murid

Pada proses ini sistem akan menampilkan daftar transaksi yang murid telah diambil oleh tutor. Pertama sistem akan mengirimkan id_user dari murid yang nantinya digunakan untuk mendapatkan daftar *history* transaksi. Kemudian melalui *volley library* data berupa *JSON* akan diterima dari *web service* yang berisi data dari *history* transaksi. Kode program dapat dilihat pada kode sumber 4.7.

```
1     private void getHistory(final String id_user) {
2         historyMuridDatas = new ArrayList<>();
3         StringRequest stringRequest = new
4     StringRequest(Request.Method.POST,
5                 Connect.HISTORYMURID, new
6     Response.Listener<String>() {
7             @Override
8             public void onResponse(String response)
9     {
10                Log.d("coba", response);
11                try {
12                    JSONObject jsonObject = new
13    JSONObject(response);
14                    if(jsonObject.length() > 0)
```

```
15                  {
16                      JSONArray arrayHistory =
17  jsonObject.getJSONArray("result");
18                      for (int i=0; i<
19  arrayHistory.length();i++)
20                      {
21                          JSONObject objectHistory
22  = arrayHistory.getJSONObject(i);
23                          HistoryMuridData
24  dataMurid = new HistoryMuridData(
25
26  objectHistory.getInt("id_history"),
27
28  objectHistory.getInt("id_pencariantutor"),
29
30  objectHistory.getString("nama_tutor"),
31
32  objectHistory.getString("telp_tutor"),
33
34  objectHistory.getString("pelajaran"),
35
36  objectHistory.getString("tanggal"),
37
38  objectHistory.getString("biaya"),
39
40  objectHistory.getString("rating"),
41
42  objectHistory.getString("komentar"));
43
44  historyMuridDatas.add(dataMurid);
45                      }
46                      mAdapter = new
47  HistoryMuridAdapter(HistoryMuridActivity.this,
48  0, historyMuridDatas);
49
50  listView.setAdapter(mAdapter);
51                  }
52                  mAdapter.notifyDataSetChanged();
53
54              } catch (JSONException e) {
55                  e.printStackTrace();
56              }
57          }
58      },
```

```
59                  new Response.ErrorListener() {
60                      @Override
61                      public void
62  onErrorResponse(VolleyError error) {
63
64  Toast.makeText(getApplicationContext(),error.get
65  Message(),Toast.LENGTH_LONG).show();
66                      }
67                  }){
68              @Override
69              protected Map<String, String>
70  getParams() throws AuthFailureError {
71                  Map<String, String> params = new
72  HashMap<>();
73                  params.put("id_user",id_user);
74                  return params;
75              }
76      };
77      RequestQueue requestQueue =
78  Volley.newRequestQueue(this);
79      requestQueue.add(stringRequest);
80  }
```

Kode Sumber 4.7 Proses Melihat *History* Transaksi Murid

4.3.6. Implementasi Proses Pemberian Rating dan Komentar

Secara fungsional proses ini digunakan bagi pengguna murid untuk memberikan rating dan komentar kepada tutor yang telah selesai melakukan transaksi pencarian tutor. Pada proses ini sistem akan menampilkan kembali informasi mulai dari nama tutor, tanggal dilaksanakannya tutor, pelajaran yang dipesan dan juga biaya transaksi tersebut. Kemudian setelah memberikan komentar dan rating, maka sistem akan mengirimkan data rating dan komentar ke basis data dengan menggunakan *volley library* melalui *web service*. Kode program dapat dilihat pada kode program 4.8.

```
1     private void rating(final int id, final float
2     rating, final String komentar) {
3         StringRequest stringRequest = new
4     StringRequest(Request.Method.POST,
5     Connect.RATING,
6                 new Response.Listener<String>() {
7                     @Override
8                     public void onResponse(String
9     response) {
10                        try {
11                            JSONObject jsonObject =
12    new JSONObject(response);
13
14    Toast.makeText(getApplicationContext(),jsonObjec
15    t.getString("message"),Toast.LENGTH_LONG).show()
16    ;
17                            finish();
18                        } catch (JSONException e) {
19                            e.printStackTrace();
20                        }
21                    }
22                },
23                new Response.ErrorListener() {
24                    @Override
25                    public void
26    onErrorResponse(VolleyError error) {
27                        error.printStackTrace();
28
29    Toast.makeText(getApplicationContext(),error.get
30    Message(),Toast.LENGTH_LONG).show();
31                    }
32                }){
33            @Override
34            protected Map<String, String>
35    getParams() throws AuthFailureError {
36                Map<String, String> params = new
37    HashMap<>();
38                params.put("id",String.valueOf(id));
39
40    params.put("rating",String.valueOf(rating));
41                params.put("komentar", komentar);
42                return params;
43            }
44        };
```

```
45      RequestQueue requestQueue =
46  Volley.newRequestQueue(this);
47      requestQueue.add(stringRequest);
48  }
```

Kode Sumber 4.8 Proses Pemberian Rating dan Komentar

4.3.7. Implementasi Proses Mengisi Ketersediaan Hari

Pada proses mengisi ketersediaan hari digunakan oleh pengguna tutor. Tutor akan memilih ketersediaan hari yang telah disediakan, mulai dari senin sampai minggu. Pilihan tutor akan dikirim ke basis data melalui *web service* dengan menggunakan *volley library*. Dan setelah data telah dimasukkan kedalam basis data, *web service* mengirimkan *JSON* yang berisikan pesan bahwa ketersediaan hari berhasil ditambah. Detail program dapat dilihat pada Kode Sumber 4.9.

```
1   public void submitHari()
2   {
3       selectedDay = new StringBuilder();
4       if (senin.isChecked()){
5           selectedDay.append("Monday,");
6       }
7       if (selasa.isChecked()){
8           selectedDay.append("Tuesday,");
9       }
10      if (rabu.isChecked()){
11          selectedDay.append("Wednesday,");
12      }
13      if (kamis.isChecked()){
14          selectedDay.append("Thusday,");
15      }
16      if (jumat.isChecked()){
17          selectedDay.append("Friday,");
18      }
19      if (sabtu.isChecked()){
20          selectedDay.append("Saturday,");
21      }
22      if (minggu.isChecked()){
```

```
23          selectedDay.append("Sunday,");
24      }
25      //Toast.makeText(this,
26  selectedDay.toString(),
27  Toast.LENGTH_SHORT).show();
28      StringRequest stringRequest = new
29  StringRequest(Request.Method.POST,
30  Connect.TAMBAHKETERSEDIAANHARI_URL,
31              new Response.Listener<String>() {
32                  @Override
33                  public void onResponse(String
34  response) {
35                      progressDialog.dismiss();
36                      try {
37                          JSONObject jsonObject =
38  new JSONObject(response);
39
40  Toast.makeText(getApplicationContext(),jsonObjec
41  t.getString("message"),Toast.LENGTH_LONG).show()
42  ;
43
44  toIntent(HomeTutorActivity.class);
45                          finish();
46                      } catch (JSONException e) {
47                          e.printStackTrace();
48                      }
49                  }
50              },
51              new Response.ErrorListener() {
52                  @Override
53                  public void
54  onErrorResponse(VolleyError error) {
55                      progressDialog.hide();
56                      error.printStackTrace();
57
58  Toast.makeText(getApplicationContext(),error.get
59  Message(),Toast.LENGTH_LONG).show();
60                  }
61              }){
62          @Override
63          protected Map<String, String>
64  getParams() throws AuthFailureError {
65              Map<String, String> params = new
66  HashMap<>();
```

```
67
68  params.put("id_user",db.getIduser());
69
70  params.put("hari",selectedDay.toString());
71              return params;
72          }
73      };
74      RequestQueue requestQueue =
75  Volley.newRequestQueue(this);
76      requestQueue.add(stringRequest);
77  }
```

Kode Sumber 4.9 Proses Mengisi Ketersediaan Hari

4.3.8. Implementasi Proses Pencarian Murid

Pada proses pencarian murid memiliki beberapa proses. Dimulai dari proses pemilihan kriteria murid yang dijadikan parameter penentuan prioritas murid yang dicari oleh tutor, setelah itu sistem baru memberikan daftar murid yang termasuk pada prioritas tutor.

4.3.8.1. Proses Penentuan Prioritas Murid

Pada proses ini tutor akan melakukan pemilihan kriteria murid digunakan untuk parameter penentuan daftar prioritas murid bagi tutor. Proses yang terjadi pada pemilihan kriteria murid ini hanyalah tutor memilih kriteria murid yang diinginkan tutor yang berguna sebagai parameter pemberian daftar prioritas murid kepada tutor. Pada halaman ini terdapat *dropdown* yang berisi kriteria murid, mulai dari berdasarkan jarak, berdasarkan pelajaran atau keahlian dari tutor, berdasarkan ketersediaan hari, berdasarkan jenis kelamin, berdasarkan kelas, berdasarkan usia, dan menampilkan semua murid. Detail program dapat dilihat pada Kode Sumber 4.10.

```
1     public void cariMurid()
2     {
3         final String kriteria;
4         if
5     (pilihKriteria.toString().equals("Berdasarkan
6     Jarak Terdekat"))
7         {
8             kriteria = "jarak";
9             Intent myIntent =  new
10    Intent(getBaseContext(),CariMuridActivity.class)
11    ;
12            Bundle bundle = new Bundle();
13            bundle.putString("kriteria", kriteria);
14            myIntent.putExtra("bundle",bundle);
15            startActivityForResult(myIntent,0);
16        }
17        else
18    if(pilihKriteria.toString().equals("Berdasarkan
19    Pelajaran/Keahlian"))
20        {
21            kriteria = "pelajaran";
22            Intent myIntent =  new
23    Intent(getBaseContext(),CariMuridActivity.class)
24    ;
25            Bundle bundle = new Bundle();
26            bundle.putString("kriteria", kriteria);
27            myIntent.putExtra("bundle",bundle);
28            startActivityForResult(myIntent,0);
29        }
30
31        else
32    if(pilihKriteria.toString().equals("Berdasarkan
33    Ketersediaan Hari"))
34        {
35            kriteria = "hari";
36            Intent myIntent =  new
37    Intent(getBaseContext(),CariMuridActivity.class)
38    ;
39            Bundle bundle = new Bundle();
40            bundle.putString("kriteria", kriteria);
41            myIntent.putExtra("bundle",bundle);
42            startActivityForResult(myIntent,0);
43        }
44        else
```

```
45  if(pilihKriteria.toString().equals("Berdasarkan
46  Jenis Kelamin"))
47      {
48          kriteria = "jenis kelamin";
49          Intent myIntent =  new
50  Intent(getBaseContext(),CariMuridActivity.class)
51  ;
52          Bundle bundle = new Bundle();
53          bundle.putString("kriteria", kriteria);
54          myIntent.putExtra("bundle",bundle);
55          startActivityForResult(myIntent,0);
56      }
57      else
58  if(pilihKriteria.toString().equals("Berdasarkan
59  Kelas"))
60      {
61          kriteria = "kelas";
62          Intent myIntent =  new
63  Intent(getBaseContext(),CariMuridActivity.class)
64  ;
65          Bundle bundle = new Bundle();
66          bundle.putString("kriteria", kriteria);
67          myIntent.putExtra("bundle",bundle);
68          startActivityForResult(myIntent,0);
69      }
70      else
71  if(pilihKriteria.toString().equals("Berdasarkan
72  Usia"))
73      {
74          kriteria = "usia";
75          Intent myIntent =  new
76  Intent(getBaseContext(),CariMuridActivity.class)
77  ;
78          Bundle bundle = new Bundle();
79          bundle.putString("kriteria", kriteria);
80          myIntent.putExtra("bundle",bundle);
81          startActivityForResult(myIntent,0);
82      }
83      else
84  if(pilihKriteria.toString().equals("Semua
85  Murid"))
86      {
87          kriteria = "all";
89          Intent myIntent =  new
```

```
90  Intent(getBaseContext(),CariMuridActivity.class)
91  ;
92          Bundle bundle = new Bundle();
93          bundle.putString("kriteria", kriteria);
94          myIntent.putExtra("bundle",bundle);
95          startActivityForResult(myIntent,0);
96      }
97  }
```

Kode Sumber 4.10 Fungsi cariMurid()

4.3.8.2. Proses Pencarian Murid

Pada proses ini terdapat dua fungsi, fungsi getMurid() dan juga fungsi getDistance(). Pada fungsi getMurid() berguna untuk manampilkan daftar prioritas murid berdasarkan kriteria yang dipilih tutor sebelumnya. Pertama *volley library* mengirimkan kriteria melalui *web service* yang berguna untuk mengambil data murid yang termasuk pada kriteria yang dipilih. Kemudian fungsi getDistance() digunakan untuk mencari jarak antara tutor dan murid serta menampilkan rute dari tutor dan murid. Detail fungsi getMurid() pada Kode Sumber 4.11 dan fungsi getDistance() pada Kode Sumber 4.12.

Fungsi getMurid() merupakan fungsi yang menerima respon dari *web service* berupa *JSON* yang berisikan data murid. Kemudian dari fungsi getMurid() akan mengirim semua data menuju ke fungsi getDistance(). Dan sebelum mengirim, pada fungsi getMurid() mengambil alamat dari tiap murid yang nantinya digunakan parameter pada fungsi getDistance().

```
1  public void getMurid()
2  {
3      progressDialog.setMessage("Please Wait
4  ...");
5      progressDialog.show();
6      StringRequest stringRequest = new
7  StringRequest(Request.Method.POST,
8  Connect.GETMURID_URL, new
```

```
9   Response.Listener<String>() {
10          @Override
11          public void onResponse(String response)
12  {
13              try {
14                  JSONObject jsonObject = new
15  JSONObject(response);
16                  if (jsonObject.length()>0)
17                  {
18                      JSONArray arrayMurid =
19  jsonObject.getJSONArray("result");
20                      for (int i=0;
21  i<arrayMurid.length();i++)
22                      {
23                          JSONObject objectMurid =
24  arrayMurid.getJSONObject(i);
25                          alamatMurid =
26  objectMurid.getString("alamat");
27
28  Log.d("ataas",alamatMurid);
29                          getDistance(objectMurid,
30  alamatMurid);
31                      }
32                  }
33
34
35                  progressDialog.dismiss();
36              } catch (JSONException e) {
37                  e.printStackTrace();
38              }
39          }
40      },
41          new Response.ErrorListener() {
42              @Override
43              public void
44  onErrorResponse(VolleyError error) {
45                  progressDialog.hide();
46
47  Toast.makeText(getApplicationContext(),error.get
48  Message(),Toast.LENGTH_LONG).show();
49              }
50          }){
51          @Override
52          protected Map<String, String>
```

```
53  getParams() throws AuthFailureError {
54              Map<String, String> params = new
55  HashMap<>();
56
57  params.put("id_user",db.getIduser());
58
59  params.put("kriteria",bundle.getString("kriteria
60  "));
61              return params;
62          }
63      };
64      RequestQueue requestQueue =
65  Volley.newRequestQueue(this);
66      requestQueue.add(stringRequest);}
```

Kode Sumber 4.11 Fungsi getMurid()

Fungsi getDistance() digunakan untuk mencari jarak antara tutor dengan masing-masing murid. Pertama fungsi ini menggunakan *API* dari *Google* untuk mengambil *longitude* dan *latitude* dari masing-masing murid. Kemudian untuk perhitungan jarak, pada aplikasi telah menyimpan alamat dari tutor, dari situ didapat pula *longitude* dan *latitude*. Kemudian didapatlah jarak antara tutor dan murid, serta rute dari alamat tutor menuju alamat murid.

```
1   public void getDistance(final JSONObject
2   objectJarak, String alamat)
3   {
4       geocoder = new Geocoder(getBaseContext());
5       try {
6           List<Address> listMurid =
7   geocoder.getFromLocationName(alamat, 1);
8           Log.d("askdjas", listMurid.toString());
9           Address alamatMurid = listMurid.get(0);
10          latMurid = alamatMurid.getLatitude();
11          longMurid = alamatMurid.getLongitude();
12
13          List<Address> listTutor =
14  geocoder.getFromLocationName(alamatTutordb, 1);
15          Log.d("tutor", listTutor.toString());
16          Address alamatTutor = listTutor.get(0);
```

```
17          latTutor = alamatTutor.getLatitude();
18          longTutor = alamatTutor.getLongitude();
19      } catch (IOException e) {
20          e.printStackTrace();
21      }
22
23      StringRequest stringRequest = new
24  StringRequest(Request.Method.GET,
25
26  "https://maps.googleapis.com/maps/api/directions
27  /json?" +
28              "origin=" +latTutor+ "," +longTutor+
29              "&destination="+latMurid+","
30  +longMurid+
31              "&key=AIzaSyCwH6FT975GOvqRVaf_-
32  rmp429uGgFXhR0", new Response.Listener<String>()
33  {
34          @Override
35          public void onResponse(String response)
36  {
37              try {
38                  JSONObject jsonObject = new
39  JSONObject(response);
40                  if (jsonObject.length()>0)
41                  {
42                      JSONArray arrayDistanceMap =
43  jsonObject.getJSONArray("routes");
44                      JSONObject objectDistanceMap
45  = arrayDistanceMap.getJSONObject(0);
46                      JSONArray jarak =
47  objectDistanceMap.getJSONArray("legs");
48                      JSONObject objectDistance =
49  jarak.getJSONObject(0);
50                      JSONObject jarakFinal =
51  objectDistance.getJSONObject("distance");
52                      getJarak =
53  Float.valueOf(jarakFinal.getString("value"));
54
55  Log.d("jarak",getJarak.toString());
56                      CariMuridData dataMurid =
57  new
58
59  CariMuridData(objectJarak.getInt("id"),
60
```

```
61  objectJarak.getInt("id_user"),
62  objectJarak.getString("name"),
63  objectJarak.getString("kelas"),
64  objectJarak.getString("pelajaran"),
65  objectJarak.getString("alamat"),
66  objectJarak.getString("tanggal"),
67  objectJarak.getString("hari"),
68  objectJarak.getString("jam"),
69  objectJarak.getString("biaya"),
70  getJarak,
71  objectJarak.getInt("durasi"));
72  cariMuridDataList.add(dataMurid);
73                  }
74                  if
75  (bundle.getString("kriteria").matches("jarak"))
76  {
77                      mAdapter.sort(new
78  Comparator<CariMuridData>() {
79                          @Override
80                          public int
81  compare(CariMuridData arg1, CariMuridData arg0)
82  {
83                              return
84  arg1.getJarak_pencarian().compareTo(arg0.getJara
85  k_pencarian());
86                          }
87                      });
88                  }
89                  mAdapter.notifyDataSetChanged();
90              } catch (JSONException e) {
91                  e.printStackTrace();
92              }
93          }
94      },
95              new Response.ErrorListener() {
96                  @Override
97                  public void
98  onErrorResponse(VolleyError error) {
99                      progressDialog.hide();
100
101 Toast.makeText(getApplicationContext(),error.get
102 Message(),Toast.LENGTH_LONG).show();
103                 }
104             });
```

```
105     RequestQueue requestQueue =
106 Volley.newRequestQueue(this);
107     requestQueue.add(stringRequest);}
```

Kode Sumber 4.12 Fungsi getDistance()

4.3.9. Implementasi Proses Melihat Keahlian

Secara fungsional proses ini digunakan pengguna sebagai tutor untuk melihat daftar keahlian yang tutor kuasai. Kode program dapat dilihat pada kode sumber 4.13. Pada fungsi getKeahlianByUsername() sistem akan mengirimkan id_user dari tutor untuk mengambil data keahlian yang dimilik. Kemudian *web service* mengirimkan umpan balik berupa *JSON* yang berisikan kelas dan pelajaran yang dikuasai tutor.

```
1   public void getKeahlianByUsername(final String
2   Id_user)
3   {
4       keahlianTutorDataList = new ArrayList<>();
5       StringRequest stringRequest = new
6   StringRequest(Request.Method.POST,
7               Connect.GETDATAKEAHLIAN_URL, new
8   Response.Listener<String>() {
9           @Override
10          public void onResponse(String response)
11  {
12              Log.d("coba", response);
13              try {
14                  JSONObject jsonObject = new
15  JSONObject(response);
16                  if(jsonObject.length() > 0)
17                  {
18                      JSONArray arrayKeahlian =
19  jsonObject.getJSONArray("result");
20                      for (int i=0; i<
21  arrayKeahlian.length();i++)
22                      {
23                          JSONObject
24  objectKeahlian = arrayKeahlian.getJSONObject(i);
25                          KeahlianTutorData
26  dataKeahlian =
```

```
27                                      new
28  KeahlianTutorData(objectKeahlian.getInt("id"),
29
30  objectKeahlian.getString("kelas"),
31
32  objectKeahlian.getString("pelajaran"));
33
34
35  keahlianTutorDataList.add(dataKeahlian);
36                      }
37
38                      mAdapter = new
39  KeahlianTutorAdapter(KeahlianTutorActivity.this,
40  0, keahlianTutorDataList);
41
42  listView.setAdapter(mAdapter);
43                  }
44                  mAdapter.notifyDataSetChanged();
45                  swipe.setRefreshing(false);
46
47              } catch (JSONException e) {
48                  e.printStackTrace();
49              }
50          }
51      },
52          new Response.ErrorListener() {
53              @Override
54              public void
55  onErrorResponse(VolleyError error) {
56
57  Toast.makeText(getApplicationContext(),error.get
58  Message(),Toast.LENGTH_LONG).show();
59              }
60          }){
61          @Override
62          protected Map<String, String>
63  getParams() throws AuthFailureError {
64              Map<String, String> params = new
65  HashMap<>();
66              params.put("id_user",Id_user);
67              return params;
68          }
69      };
70      RequestQueue requestQueue =
```

```
71  Volley.newRequestQueue(this);
72      requestQueue.add(stringRequest);
73  }
```

Kode Sumber 4.13 Proses Melihat Keahlian

4.3.10. Implementasi Proses Menambah Keahlian

Secara fungsional proses menambah keahlian dapat digunakan oleh pengguna sebagai tutor. Pada proses menambah keahlian, tutor akan mengisi kelas dan pelajaran yang tutor kuasai, kemudia *volley library* akan mengirimkan data tersebut ke basis data melalui *web service*. Kode program dapat dilihat pada kode sumber 4.14.

```
1   private void tambahKeahlian()
2   {
3       final String stringKelas, stringPelajaran;
4       stringKelas = pilihKelas;
5       stringPelajaran =
6   pelajaran.getText().toString();
7       progressDialog.setMessage("Tambah
8   Keahlian...");
9       progressDialog.show();
10      StringRequest stringRequest = new
11  StringRequest(Request.Method.POST,
12  Connect.TAMBAHKEAHLIAN_URL,
13              new Response.Listener<String>() {
14                  @Override
15                  public void onResponse(String
16  response) {
17                      progressDialog.dismiss();
18                      try {
19                          JSONObject jsonObject =
20  new JSONObject(response);
21
22  Toast.makeText(getApplicationContext(),jsonObjec
23  t.getString("message"),Toast.LENGTH_LONG).show()
24  ;
25                          finish();
26                      } catch (JSONException e) {
27                          e.printStackTrace();
```

```
28                      }
29                  }
30              },
31              new Response.ErrorListener() {
32                  @Override
33                  public void
34  onErrorResponse(VolleyError error) {
35                      progressDialog.hide();
36                      error.printStackTrace();
37
38  Toast.makeText(getApplicationContext(),error.get
39  Message(),Toast.LENGTH_LONG).show();
40                  }
41              }){
42          @Override
43          protected Map<String, String>
44  getParams() throws AuthFailureError {
45              Map<String, String> params = new
46  HashMap<>();
47
48  params.put("id_user",db.getIduser());
49              params.put("kelas",stringKelas);
50
51  params.put("pelajaran",stringPelajaran);
52              return params;
53          }
54      };
55      RequestQueue requestQueue =
56  Volley.newRequestQueue(this);
57      requestQueue.add(stringRequest);
58  }
```

Kode Sumber 4.14 Proses Menambah Keahlian

4.3.11. Implementasi Proses Melihat *Profile* Tutor

Proses ini terjadi bagi pengguna tutor untuk melihat data diri mereka. Pada fungsi getProfileTutor() sistem mengirimkan id_user yang berguna untuk mengambil data dari basis data sesuai dengan id_user tersebut. Kode program dapat dilihat pada kode sumber 4.15.

```
1   public void getProfileTutor(final String
2   id_user)
3   {
4       StringRequest stringRequest = new
5   StringRequest(Request.Method.POST,
6               Connect.PROFILE_URL, new
7   Response.Listener<String>() {
8           @Override
9           public void onResponse(String response)
10  {
11              try {
12                  JSONObject jsonObject = new
13  JSONObject(response);
14                  JSONObject arrayKeahlian =
15  jsonObject.getJSONObject("result");
16                  JSONArray arrayUser =
17  arrayKeahlian.getJSONArray("user");
18                  JSONObject objectProfile =
19  arrayUser.getJSONObject(0);
20                  nama =
21  objectProfile.getString("nama_user");
22                  alamat =
23  objectProfile.getString("alamat_user");
24                  notelp=
25  objectProfile.getString("telp_user");
26                  email=
27  objectProfile.getString("email_user");
28
29                  JSONArray arrayHari =
30  arrayKeahlian.getJSONArray("hari");
31                  hari="";
32                  for (int i=0; i<
33  arrayHari.length();i++)
34                  {
35                      JSONObject objectHari =
36  arrayHari.getJSONObject(i);
37                      hari = hari+"
38  "+objectHari.getString("hari_tutor");
39                  }
40
41  setView(nama,alamat,notelp,email, hari);
42
43              } catch (JSONException e) {
44                  e.printStackTrace();
```

```
45                }
46            }
47        },
48                new Response.ErrorListener() {
49                    @Override
50                    public void
51    onErrorResponse(VolleyError error) {
52
53    Toast.makeText(getApplicationContext(),error.get
54    Message(),Toast.LENGTH_LONG).show();
55                    }
56                }){
57            @Override
58            protected Map<String, String>
59    getParams() throws AuthFailureError {
60                Map<String, String> params = new
61    HashMap<>();
62                params.put("id_user",id_user);
63                params.put("jenis",jenis);
64                return params;
65            }
66        };
67        RequestQueue requestQueue =
68    Volley.newRequestQueue(this);
69        requestQueue.add(stringRequest);
70    }
```

Kode Sumber 4.15 Proses Melihat *Profile* Tutor

4.3.12. Implementasi Proses Mengubah *Profile* Tutor

Pada proses ini, tutor akan mengisi formulir yang sudah tersedia. Pada formulir ini sistem menampilkan data yang belum terganti supaya tutor hanya mengganti data diri yang ingin diganti saja. Kemudian setelah mengisi formulir, maka *volley library* akan mengirimkan data ke basis data melalui *web service*. Kode program dapat dilihat pada kode sumber 4.16.

```
1     private void editProfile() {
2
3         int count = 0;
4
5         nama = etnama.getText().toString();
```

```
6       telp = ettelp.getText().toString();
7       email = etemail.getText().toString();
8
9       selectedDay = new StringBuilder();
10      if (senin.isChecked()) {
11          selectedDay.append("Senin,");
12          count = 0;
13      }
14      if (selasa.isChecked()) {
15          selectedDay.append("Selasa,");
16          count = 0;
17      }
18      if (rabu.isChecked()) {
19          selectedDay.append("Rabu,");
20          count = 0;
21      }
22      if (kamis.isChecked()) {
23          selectedDay.append("Kamis,");
24          count = 0;
25      }
26      if (jumat.isChecked()) {
27          selectedDay.append("Jumat,");
28          count = 0;
29      }
30      if (sabtu.isChecked()) {
31          selectedDay.append("Sabtu,");
32          count = 0;
33      }
34      if (minggu.isChecked()) {
35          selectedDay.append("Minggu,");
36          count = 0;
37      }
38      if (selectedDay.toString().equals("")) {
39          Toast.makeText(this, "Minimal 1 hari",
40  Toast.LENGTH_SHORT).show();
41          count = 1;
42      }
43      if (count == 0) {
44
45          progressDialog.setMessage("Ubah
46  Profil...");
47          progressDialog.show();
48          StringRequest stringRequest = new
49  StringRequest(Request.Method.POST,
```

```
50  Connect.EDITPROFILE_URL,
51                  new Response.Listener<String>()
52  {
53                      @Override
54                      public void
55  onResponse(String response) {
56
57  progressDialog.dismiss();
58                          try {
59                              JSONObject
60  jsonObject = new JSONObject(response);
61
62  Toast.makeText(getApplicationContext(),
63  jsonObject.getString("message"),
64  Toast.LENGTH_LONG).show();
65
66  db.updateAlamat(username,alamat);
67
68                              finish();
69                          } catch (JSONException
70  e) {
71                              e.printStackTrace();
72                          }
73                      }
74                  },
75                  new Response.ErrorListener() {
76                      @Override
77                      public void
78  onErrorResponse(VolleyError error) {
79                          progressDialog.hide();
80                          error.printStackTrace();
81
82  Toast.makeText(getApplicationContext(),
83  error.getMessage(), Toast.LENGTH_LONG).show();
84                          }
85                  }) {
86              @Override
87              protected Map<String, String>
88  getParams() throws AuthFailureError {
89                  Map<String, String> params = new
90  HashMap<>();
91                  params.put("id_user",
92  db.getIduser());
93                  params.put("nama", nama);
```

```
94                  params.put("alamat", alamat);
95                  params.put("telp", telp);
96                  params.put("email", email);
97                  params.put("ketersediaanhari",
98  selectedDay.toString());
99                  return params;
100             }
101         };
102         RequestQueue requestQueue =
103 Volley.newRequestQueue(this);
104         requestQueue.add(stringRequest);
105     }
106 }
```

Kode Sumber 4.16 Proses Mengubah *Profile* Tutor

4.3.13. Implementasi Proses Melihat *History* Transaksi Tutor

Pada proses ini sistem akan menampilkan daftar transaksi yang telah berhasil milik tutor. Pertama sistem akan mengirimkan id_user dari tutor yang nantinya digunakan untuk mendapatkan daftar *history* transaksi. Kemudian melalui *volley library* data berupa *JSON* akan diterima dari *web service* yang berisi data dari *history* transaksi. Kode program dapat dilihat pada kode sumber 4.17.

```
1   private void getHistory(final String username) {
2       StringRequest stringRequest = new
3   StringRequest(Request.Method.POST,
4               Connect.HISTORYTUTOR, new
5   Response.Listener<String>() {
6           @Override
7           public void onResponse(String response)
8   {
9               Log.d("coba", response);
10              try {
11                  JSONObject jsonObject = new
12  JSONObject(response);
13                  if(jsonObject.length() > 0)
14                  {
15                      JSONArray arrayHistory =
```

```
16  jsonObject.getJSONArray("result");
17                      for (int i=0; i<
18  arrayHistory.length();i++)
19                      {
20                          JSONObject objectHistory
21  = arrayHistory.getJSONObject(i);
22                          HistoryTutorData
23  dataTutor = new HistoryTutorData(
24
25  objectHistory.getInt("id_history"),
26
27  objectHistory.getString("tanggal"),
28
29  objectHistory.getString("rating"),
30
31  objectHistory.getString("komentar"));
32
33  historyTutorDataList.add(dataTutor);
34                      }
35                  }
36                  mAdapter.notifyDataSetChanged();
37
38              } catch (JSONException e) {
39                  e.printStackTrace();
40              }
41          }
42      },
43              new Response.ErrorListener() {
44                  @Override
45                  public void
46  onErrorResponse(VolleyError error) {
47
48  Toast.makeText(getApplicationContext(),error.get
49  Message(),Toast.LENGTH_LONG).show();
50                  }
51              }){
52          @Override
53          protected Map<String, String>
54  getParams() throws AuthFailureError {
55              Map<String, String> params = new
56  HashMap<>();
57              params.put("username",username);
58              return params;
59          }
```

```
60      };
61      RequestQueue requestQueue =
62  Volley.newRequestQueue(this);
63      requestQueue.add(stringRequest);
64  }
```

Kode Sumber 4.17 Proses Melihat *History* Transaksi Tutor

[Halaman ini sengaja dikosongkan]

BAB V
PENGUJIAN DAN EVALUASI

Pada bab ini dijelaskan pengujian dan evaluasi dari aplikasi FindingTutor. Pengujian yang dilakukan adalah pengujian terhadap kebutuhan fungsionalitas sistem yang telah dijabarkan pada Bab III. Hasil evaluasi menjabarkan tentang rangkuman hasil pengujian pada bagian akhir bab ini.

5.1. Lingkungan Pengujian

Lingkungan pengujian sistem pada pengerjaan Tugas Akhir ini dilakukan pada lingkungan dan alat kakas sebagai berikut:

- Perangkat Bergerak A
  Tipe : Redmi Note 3
  Versi Android : 5.0.2 LRX22G
  Prosesor : Octa-core (2.0 GHz)
  RAM : 3GB
  Memori Internal : 32GB

- Perangkat Bergerak B
  Tipe : MI 4LTE
  Versi Android : 6.0.1 MMB29M
  Prosesor : Quad-core Max 2.5GHz
  RAM : 3GB
  Memori Internal : 16GB

5.2. Skenario Pengujian

Pada bagian ini akan dijelaskan tentang skenario pengujian yang dilakukan pada aplikasi FindingTutor. Pengujian yang dilakukan adalah pengujian kebutuhan fungsionalitas. Pengujian dilakukan oleh sepuluh orang yang mana lima orang akan berperan sebagai tutor dan lima orang akan berperan sebagai murid.

5.2.1. Pengujian Fungsionalitas

Pengujian fungsionalitas adalah pengujian kebutuhan fungsional yang dilakukan dengan menggunakan metode *balck box*. Metode *black box* adalah metode di mana pengujian ditekankan pada pola masukan dan keluaran yang sesuai dengan skenario. Pengujian yang dilakukan mengacu pada kasus penggunaan yang dijelaskan pada Bab III.

Pengujian fungsionalitas aplikasi dilakukan secara mandiri dengan melakukan skenario yang sama dengan rancangan alur proses aplikasi sebagai tolok ukur keberhasilan pengujian, dan mengacu pada kasus penggunaan yang sebelumnya telah dijelaskan pada Bab III. Pengujian pada kebutuhan fungsionalitas dapat dijabarkan pada subbab berikut.

5.2.1.1. Pengujian Mencari Tutor

Pada pengujian ini pengguna sebagai murid akan melakukan pencarian tutor. Skenario pengujian pencarian tutor dapat dilihat pada Tabel 5.1. Hasil pengujian dari proses pencarian tutor dapat dilihat pada Gambar 4.4 dan Gambar 4.5. Hasil pengujian pemberian kriteria dapat dilihat pada Gambar 4.3. Hasil pengujian estimasi biaya dapat dilihat pada Gambar 4.6.

Tabel 5.1 Pengujian Mencari Tutor

ID	SP-0001
Kasus Penggunaan	Melakukan pencarian tutor
Tujuan Pengujian	Menguji fitur *Find tutor*
Skenario	Murid melakukan pencarian tutor
Kondisi Awal	Murid memilih menu *find tutor*
Langkah Pengujian	1. Murid memilih menu *find tutor* 2. Murid menggunakan fitur kriteria otomatis 3. Murid mengisi formulir pencarian tutor 4. Murid menyetujui estimasi biaya yang diberikan

Hasil yang Diharapkan	Sistem memberikan pesan pencarian telah berhasil
Hasil yang Didapat	Sistem berhasil menampilkan pesan pencarian telah berhasil
Hasil Pengujian	Berhasil

5.2.1.2. Pengujian Melihat Transaksi Sedang Berjalan

Pada pengujian ini pengguna sebagai murid akan melihat transaksi yang sedang berjalan. Pada pengujian ini, pengguna sebagai murid dapat melihat proses transaksi yang mereka pesan telah diambil oleh tutor. Skenario pengujian dapat dilihat pada Tabel 5.2 dan hasil pengujian dapat dilihat pada Gambar 4.7 dan Gambar 4.8.

Tabel 5.2 Pengujian Transaksi Sedang Berjalan

ID	SP-0002
Kasus Penggunaan	Melihat transaksi sedang berjalan
Tujuan Pengujian	Menguji fitur *On Process Transaction*
Skenario	Murid melihat daftar pencarian tutor yang sedang berlangsung
Kondisi Awal	Murid memilih menu *on process transaction*
Langkah Pengujian	1. Murid memilih menu *on process transaction* 2. Murid melihat daftar transaksi yang sedang berjalan 3. Murid menekan tombol detail 4. Murid melihat informasi tutor dan transaksi yang sedang berjalan
Hasil yang Diharapkan	Sistem menampilkan detail dari setiap transaksi yang sedang berjalan
Hasil yang Didapat	Sistem berhasil menampilkan detail dari setiap transaksi yang sedang berjalan
Hasil Pengujian	Berhasil

5.2.1.3. Pengujian Melihat *Profile* Murid

Pada pengujian ini pengguna sebagai murid akan melihat *profile*. Skenario pengujian dapat dilihat pada Tabel 5.3 dan hasil pengujian melihat *profile* murid pada Gambar 4.9.

Tabel 5.3 Pengujian Melihat *Profile* Murid

ID	SP-0003
Kasus Penggunaan	Melihat *profile* murid
Tujuan Pengujian	Menguji fitur *Profile* murid
Skenario	Murid melihat data diri
Kondisi Awal	Murid memilih menu *profile*
Langkah Pengujian	1. Murid memilih menu *profile* 2. Murid melihat data diri
Hasil yang Diharapkan	Sistem menampilkan data diri dari murid
Hasil yang Didapat	Sistem berhasil menampilkan data diri dari murid
Hasil Pengujian	Berhasil

5.2.1.4. Pengujian Mengubah *Profile* Murid

Pada pengujian ini pengguna sebagai murid dapat melakukan perubahand *profile*. Skenario dapat dilihat pada Tabel 5.4 dan hasil pengujian mengubah *profile* murid dapat dilihat pada Gambar 4.10.

Tabel 5.4 Pengujian Mengubah *Profile* Murid

ID	SP-0004
Kasus Penggunaan	Melihat *profile*
Tujuan Pengujian	Menguji fitur mengubah *Profile*
Skenario	Murid mengubah data diri
Kondisi Awal	Murid menekan tombol *edit* pada halaman *profile*
Langkah Pengujian	1. Murid menekan tombol *edit*

	2. Murid mengisi formulir *edit profile* 3. Murid dan tutor menekan tombol *submit*
Hasil yang Diharapkan	Data diri murid berubah
Hasil yang Didapat	Data diri murid berhasil berubah sesuai dengan data yang diisikan pada formulir
Hasil Pengujian	Berhasil

5.2.1.5. Pengujian Melihat *History* Transaksi Murid

Pada pengujian ini pengguna sebagai murid dapat melihat *history* transaksi yang telah mereka lakukan. Skenario dapat dilihat pada Tabel 5.5 dan hasil pengujian melihat *history* transaksi murid dapat dilihat pada Gambar 4.11.

Tabel 5.5 Pengujian Melihat *History* Transaksi

ID	SP-0005
Kasus Penggunaan	Melihat *history* transaksi murid
Tujuan Pengujian	Menguji fitur *History*
Skenario	Murid melihat daftar *history* transaksi yang pernah dilakukan
Kondisi Awal	Murid memilih menu *history*
Langkah Pengujian	1. Murid memilih menu *history* 2. Murid melihat daftar *history* transaksi
Hasil yang Diharapkan	Sistem menampilkan daftar *history* transaksi yang pernah murid lakukan
Hasil yang Didapat	Sistem berhasil menampilkan daftar *history* transaksi yang pernah murid lakukan
Hasil Pengujian	Berhasil

5.2.1.6. Pengujian Memberikan Rating dan Komentar

Pada pengujian ini pengguna sebagai murid dapat memberikan rating dan komentar kepada tutor setelah semua transaksi selesai. Skenario pengujian dapat dilihat pada Tabel 5.6 dan hasil pengujian dapat dilihat pada Gambar 4.12.

Tabel 5.6 Pengujian Memberikan Rating dan Komentar

ID	SP-0006
Kasus Penggunaan	Memberikan rating dan komentar
Tujuan Pengujian	Menguji fitur rating dan komentar
Skenario	Murid memberikan rating dan komentar kepada tutor
Kondisi Awal	Murid menekan tombol rating
Langkah Pengujian	1. Murid menekan tombol rating pada daftar 2. Murid melihat data transaksi 3. Murid mengisi formulir rating dan komentar 4. Murid menekan tombol *submit*
Hasil yang Diharapkan	Rating dan komentar ditambahkan, sistem menampilkan pesan data berhasil ditambah
Hasil yang Didapat	Rating dan komentar berhasil ditambah, sistem menampilkan pesan data berhasil ditambah
Hasil Pengujian	Berhasil

5.2.1.7. Pengujian Mengisi Ketersediaan Hari

Pada pengujian ini pengguna sebagai tutor akan mengisikan ketersediaan hari yang berguna sebagai salah satu kriteria untuk mencari murid. Skenario pengujian dapat dilihat pada Tabel 5.7. hasil pengujian dapat dilihat pada Gambar 4.13.

Tabel 5.7 Pengujian Mengisi Ketersediaan Hari

ID	SP-0007
Kasus Penggunaan	Mengisi ketersediaan hari
Tujuan Pengujian	Menguji fitur ketersediaan hari
Skenario	Tutor mengisikan ketersediaan hari yang akan digunakan sebagai salah satu kriteria mencari murid
Kondisi Awal	Tutor telah *login*
Langkah Pengujian	1. Pengguna telah *login* sebagai tutor 2. Tutor memilih ketersediaan hari 3. Tutor menekan tombol *accept*
Hasil yang Diharapkan	Sistem menampilkan pesan ketersediaan hari telah diisi
Hasil yang Didapat	Sistem berhasil menampilkan pesan ketersediaan hari telah diisi
Hasil Pengujian	Berhasil

5.2.1.8. Pengujian Mencari Murid

Pada pengujian ini pengguna sebagai tutor akan melakukan pencarian murid. Skenario pengujian dapat dilihat pada Tabel 5.8. Hasil pengujian dari pemilihan kriteria murid dapat dilihat pada Gambar 4.15, dan hasil pengujian daftar murid dan pengujian detail murid dapat dilihat pada Gambar 4.16 dan Gambar 4.17.

Tabel 5.8 Pengujian Mencari Murid

ID	SP-0008
Kasus Penggunaan	Mencari Murid
Tujuan Pengujian	Menguji fitur *find student*
Skenario	Tutor mencari murid berdasarkan kriteria yang dipilih
Kondisi Awal	Tutor memilih menu *find student*
Langkah Pengujian	1. Tutor memilih menu *find student* 2. Tutor memilih kriteria murid yang ingin ditampilkan 3. Tutor melihat daftar prioritas murid

Hasil yang Diharapkan	Tutor melihat daftar murid sesuai dengan kriteria yang dipilih
Hasil yang Didapat	Tutor berhasil melihat daftar murid sesuai dengan kriteria yang dipilih
Hasil Pengujian	Berhasil

5.2.1.9. Pengujian Melihat Keahlian

Pada pengujian ini pengguna sebagai tutor dapat melihat keahlian yang mereka punya. Skenario pengujian dapat dilihat pada Tabel 5.9 dan hasil pengujian dapat dilihat pada Gambar 4.18.

Tabel 5.9 Pengujian Melihat Keahlian

ID	SP-0009
Kasus Penggunaan	Melihat Keahlian
Tujuan Pengujian	Menguji fitur *skill*
Skenario	Tutor melihat daftar keahlian yang dimiliki
Kondisi Awal	Tutor memilih menu *skill*
Langkah Pengujian	1. Tutor memilih menu *skill* 2. Tutor melihat daftar keahlian yang dimiliki
Hasil yang Diharapkan	Tutor melihat daftar keahlian
Hasil yang Didapat	Tutor berhasil melihat daftar keahlian
Hasil Pengujian	Berhasil

5.2.1.10. Pengujian Menambah Keahlian

Pada pengujian ini pengguna sebagai tutor dapat menambahkan keahlian mereka. Skenario pengujian dapat dilihat pada Tabel 5.10 dan hasil pengujian dapat dilihat pada Gambar 4.19.

Tabel 5.10 Pengujian Menambah Keahlian

ID	SP-0010
Kasus Penggunaan	Menambah Keahlian

Tujuan Pengujian	Menguji fitur *add skill*
Skenario	Tutor menambahk keahlian untuk dijadikan kriteria pencarian murid
Kondisi Awal	Tutor menekan tombol *add*
Langkah Pengujian	1. Tutor menekan tombol *add* 2. Tutor memilih kelas yang dikuasai 3. Tutor mengisi pelajaran yang dikuasai 4. Tutor menekan tombol *submit*
Hasil yang Diharapkan	Data keahlian tutor bertambah
Hasil yang Didapat	Tutor berhasil menambahkan data keahlian
Hasil Pengujian	Berhasil

5.2.1.11. Pengujian Melihat *Profile* Tutor

Pada pengujian ini pengguna sebagai tutor akan melihat *profile*. Skenario pengujian dapat dilihat pada Tabel 5.11 dan hasil pengujian melihat *profile* tutor pada Gambar 4.20.

Tabel 5.11 Pengujian Melihat *Profile* Tutor

ID	SP-0011
Kasus Penggunaan	Melihat *profile* Tutor
Tujuan Pengujian	Menguji fitur *Profile* tutor
Skenario	Tutor melihat data diri
Kondisi Awal	Tutor memilih menu *profile*
Langkah Pengujian	1. Tutor memilih menu *profile* 2. Tutor melihat data diri
Hasil yang Diharapkan	Sistem menampilkan data diri dari tutor
Hasil yang Didapat	Sistem berhasil menampilkan data diri dari tutor
Hasil Pengujian	Berhasil

5.2.1.12. Pengujian Mengubah *Profile* Tutor

Pada pengujian ini pengguna sebagai tutor dapat melakukan perubahan *profile*. Skenario dapat dilihat pada Tabel 5.12 dan hasil pengujian mengubah *profile* tutor dapat dilihat pada Gambar 4.21.

Tabel 5.12 Pengujian Mengubah *Profile* Tutor

ID	SP-0012
Kasus Penggunaan	Melihat *profile*
Tujuan Pengujian	Menguji fitur mengubah *Profile*
Skenario	Murid dan tutor mengubah data diri
Kondisi Awal	Murid dan tutor menekan tombol *edit* pada halaman *profile*
Langkah Pengujian	1. Murid dan tutor menekan tombol *edit* 2. Murid dan tutor mengisi formulir *edit profile* 3. Murid dan tutor menekan tombol *submit*
Hasil yang Diharapkan	Data diri murid dan tutor berubah
Hasil yang Didapat	Data diri murid dan tutor berhasil berubah sesuai dengan data yang diisikan pada formulir
Hasil Pengujian	Berhasil

5.2.1.13. Pengujian Melihat *History* Transaksi Tutor

Pada pengujian ini pengguna sebagai tutor dapat melihat *history* transaksi yang telah mereka lakukan. Skenario dapat dilihat pada Tabel 5.13 dan hasil pengujian dan hasil pengujian melihat *history* transaksi tutor dapat dilihat pada Gambar 4.22.

Tabel 5.13 Pengujian Melihat *History* Transaksi Tutor

ID	SP-0013
Kasus Penggunaan	Melihat *history* transaksi tutor
Tujuan Pengujian	Menguji fitur *History*
Skenario	Tutor melihat daftar *history* transaksi yang pernah dilakukan
Kondisi Awal	Tutor memilih menu *history*
Langkah Pengujian	1. Tutor memilih menu *history* 2. Tutor melihat daftar *history* transaksi

Hasil yang Diharapkan	Sistem menampilkan daftar *history* transaksi yang pernah tutor lakukan
Hasil yang Didapat	Sistem berhasil menampilkan daftar *history* transaksi yang pernah tutor lakukan
Hasil Pengujian	Berhasil

5.2.2. Pengujian Ketertarikan Partisipan terhadap Aplikasi

Selain pengujian yang dilakukan untuk melihat kesesuaian masukan dengan keluaran lewat pengujian fungsional, dilakukan pengujian kepada pengguna untuk mengetahui seberapa besar ketertarikan partisipan terhadap aplikasi.

Pengujian dilakukan pada sepuluh orang yang mana lima orang akan menjadi tutor dan lima orang akan menjadi murid. Uji coba yang dilakukan partisipan meliputi melakukan pencarian tutor, melihat transaksi sedang berjalan, melihat *history* transaksi, memberikan rating dan komentar, dan mencari murid. Daftar partisipan dan hasil kuesioner dapat dilihat pada Tabel 5.14, Tabel 5.15 dan Tabel 5.16.

Tabel 5.14 Daftar Partisipan

No	Nama	Pengguna
1	Ilham M. Misbahuddin	Murid
2	Rizky Fenaldo M	Tutor
3	Nadia Rahmatin	Murid
4	Tiara Anggita	Tutor
5	M. Farhan Maulan	Tutor
6	Yohana Desy P	Murid
7	Raras Anggita	Tutor
8	Huda Fauzan M	Murid
9	Abdul Majid H	Tutor
10	M Buyung Abiyoso	Murid
11	Desy Rahmi	Murid
12	Kania Amalia	Tutor

13	Muhammad Divi Jaya	Murid
14	Putro Satrio	Tutor

Tabel 5.15 Hasil Kuesioner Pengguna Murid

No	Pertanyaan	Sangat Setuju	Setuju	Kurang Setuju	Tidak Setuju
1	Apakah tampilan aplikasi mempermudah dan membuat anda nyaman dalam menggunakan aplikasi?	15%	85%	0%	0%
2	Apakah aplikasi memberikan kenyamanan dan kemudahan saat melakukan pencarian tutor?	0	100%	0%	0%
3	Apakah fitur pemberian kriteria tutor secara otomatis sesuai dengan kriteria anda?	15%	85%	0%	0%
4	Apakah fitur *on process transaction* membantu anda dalam memberikan informasi tutor yang mengambil pesanan anda?	34%	66%	0%	0%
5	Apakah fitur pemberian rating dan komentar dapat digunakan sebagai acuan penilaian kinerja tutor?	34%	66%	0%	0%

Tabel 5.16 Hasil Kuesioner Pengguna Tutor

No	Pertanyaan	Sangat Setuju	Setuju	Kurang Setuju	Tidak Setuju
1	Apakah fitur pemilihan kriteria membantu anda mendapatkan prioritas murid yang sesuai dengan kriteria dipilih?	66%	34%	0%	0%
2	Apakah daftar prioritas murid yang diberikan sesuai dengan kriteria yang anda berikan?	34%	66%	0%	0%
3	Apakah fitur detail murid membantu anda mendapatkan informasi tentang murid?	66%	34%	0%	0%
4	Apakah jalur yang diberikan pada peta sudah sesuai dengan tempat anda dan tempat murid?	34%	66%	0%	0%
5	Apakah fitur *history* dapat digunakan sebagai acuan penilaian kinerja anda?	34%	66%	0%	0%

5.3. Evaluasi Pengujian

Pada subbab ini akan diberikan hasil evaluasi dari pengujian-pengujian yang telah dilakukan. Evaluasi yang diberikan meliputi evaluasi pengujian kebutuhan fungsional dan evaluasi pengujian ketertarikan partisipan terhadap aplikasi.

5.3.1. Evaluasi Pengujian Fungsionalitas

Hasil pengujian fungsionalitas secara keseluruhan dapat dilihat pada Tabel 5.17. berdasarkan data pada tabel tersebut, semua skenario pengujian berhasil dan program berjalan dengan baik. Sehingga dapat ditarik kesimpulan bahwa fungsionalitas dari aplikasi bekerja sesuai dengan yang diharapkan.

Tabel 5.17 Hasil Pengujian Fungsionalitas

ID	Nama	Hasil
SP-0001	Pengujian Mencari Tutor	Berhasil
SP-0002	Pengujian Melihat Transaksi Sedang Berjalan	Berhasil
SP-0003	Pengujian Melihat *Profile* Murid	Berhasil
SP-0004	Pengujian Mengubah *Profile* Murid	Berhasil
SP-0005	Pengujian Melihat *History* Transaksi Murid	Berhasil
SP-0006	Pengujian Memberikan Rating dan Komentar	Berhasil
SP-0007	Pengujian Mengisi Ketersediaan Hari	Berhasil
SP-0008	Pengujian Mencari Murid	Berhasil
SP-0009	Pengujian Melihat Keahlian	Berhasil
SP-0010	Pengujian Menambah Keahlian	Berhasil
SP-0011	Pengujian Melihat *Profile* Tutor	Berhasil
SP-0012	Pengujian Mengubah *Profile* Tutor	Berhasil
SP-0013	Pengujian Melihat *History* Transaksi Tutor	Berhasil

5.3.2. Evaluasi Pengujian Ketertarikan Partisipan terhadap Aplikasi

Berdasarkan hasil kuesioner pada Tabel 5.15 dan Tabel 5.16, dapat ditarik kesimpulan bahwa aplikasi ini dapat membantu partisipan dalam melakukan pencarian tutor atau guru les. Hal ini dapat dilihat dari 100% setuju bahwa dengan aplikasi ini dapat memberikan kemudahan dan kenyamanan untuk melakukan pencarian tutor. Selain itu dengan aplikasi partisipan tidak perlu lagi memasukkan tutor yang diingingkan dengan adanya fitur pemberian kriteria tutor.

Selain itu, aplikasi ini dapat memberikan prioritas kepada partisipan untuk melakukan pencarian murid sesuai dengan keadaannya. Hal itu dapat dilihat dari 66% sangat setuju dan 36%

setuju dengan fitur pemilihan kriteria dan 66% setuju dan 36% sangat setuju dengan daftar prioritas murid yang diberikan setelah memilih kriteria.

[Halaman ini sengaja dikosongkan

BAB VI
KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diperoleh selama pengerjaan tugas akhir dan saran mengenai pengembangan yang dapat dilakukan terhadap tugas akhir ini di masa yang akan datang.

6.1. Kesimpulan

Dari hasil pengamatan selama proses perancangan, implementasi, dan pengujian perangkat lunak yang dilakukan, dapat diambil kesimpulan sebagai berikut:

1. Aplikasi Finding-Tutor mampu memberikan kriteria tutor secara otomatis kepada murid yang ingin melakukan pencarian tutor.
2. Aplikasi Finding-Tutor mampu memberikan tutor atau guru les sesuai dengan kriteria yang diinginkan murid
3. Aplikasi Finding-Tutor dapat memberikan daftar prioritas murid kepada tutor atau guru les.
4. Proses bisnis yang terjadi antara murid dan tutor dalam melakukan pencarian tutor dan pencarian murid adalah sebagai berikut:
   a. Murid mengisi formulir pencarian tutor
   b. Murid menyetujui estimasi harga yang diberikan sistem
   c. Tutor memilih kriteria murid yang ingin dicari
   d. Tutor memilih murid yang sesuai dengan keadaannya
   e. Murid dapat melihat transaksi yang sudah diambil oleh tutor.
5. Aplikasi Finding-Tutor sudah memenuhi kebutuhan fungsional yang didefinisakn sebelumnya dengan pengujian *blackbox*.

6.2. Saran

Berikut merupakan beberapa saran untuk pengembangan sistem dimasa yang akan datang. Saran-saran ini didasarkan pada hasil perancangan, implementasi dan pengujian yang telah dilakukan.

1. Penambahan fitur notifikasi bagi pengguna murid ketika transaksi berhasil diambil oleh tutor.
2. Penambahan menu bantuan yang disediakan bagi pengguna awam.

DAFTAR PUSTAKA

[1] "Data Referensi Pendidikan." [Online]. Available: http://referensi.data.kemdikbud.go.id/index11.php?kode=056000&level=2. [Accessed: 05-May-2017].

[2] "Arti kata tutor - Kamus Besar Bahasa Indonesia (KBBI) Online." [Online]. Available: http://kbbi.web.id/tutor. [Accessed: 05-May-2017].

[3] "Android," *Android*. [Online]. Available: https://www.android.com/intl/id_id/. [Accessed: 05-May-2017].

[4] I. Rusman, "Sejarah dan Perkembangan Android Dari Masa ke masa," *Indravedia Blog*. .

[5] "Adding Maps | Android Developers." [Online]. Available: https://developer.android.com/training/maps/index.html. [Accessed: 05-May-2017].

[6] "Panduan Developer | Google Maps Geocoding API," *Google Developers*. [Online]. Available: https://developers.google.com/maps/documentation/geocoding/intro?hl=id. [Accessed: 05-May-2017].

[7] "Memulai | Google Maps Directions API," *Google Developers*. [Online]. Available: https://developers.google.com/maps/documentation/directions/start?hl=id. [Accessed: 05-May-2017].

[8] "Memulai | Google Maps Distance Matrix API," *Google Developers*. [Online]. Available: https://developers.google.com/maps/documentation/distance-matrix/start?hl=id. [Accessed: 05-May-2017].

[9] "Place Autocomplete | Google Places API for Android," *Google Developers*. [Online]. Available: https://developers.google.com/places/android-api/autocomplete?hl=id. [Accessed: 05-May-2017].

[10] "An Introduction to Volley," *Code Envato Tuts+*. [Online]. Available: https://code.tutsplus.com/tutorials/an-introduction-to-volley--cms-23800. [Accessed: 05-May-2017].

[11]"Google Trends.html." .
[12]Y. Shulin and H. Jieping, "Research and implementation of Web Services in Android network communication framework Volley," in *2014 11th International Conference on Service Systems and Service Management (ICSSSM)*, 2014, pp. 1–3.
[13]"11 Best PHP Frameworks for Modern Web Developers in 2017," *Coders Eye - Web Dev Tutorials and How-To Guides for Beginners*, 28-Sep-2016. .
[14]"Codeigniter.pdf." .
[15]"JSON." [Online]. Available: http://www.json.org/json-id.html. [Accessed: 08-May-2017].

BIODATA PENULIS

Penulis, **Riska Adhita**, lahir di Pati, 20 Maret 1995. Penulis adalah anak kedua dari dua bersaudara. Penulis menempuh pendidikan sekolah dasar di SD Rajawali Juwana. Melanjutkan pendidikan sekolah menengah pertama di SMP Negeri 3 Pati dan penulis menempuh pendidikan menengah atas di SMA Negeri 1 Pati. Selanjutnya penulis melanjutkan pendidikan sarjana di Jurusan Teknik Informatika, Fakultas Teknologi dan Informasi, Institut Teknologi Sepuluh Nopember Surabaya. Selama kuliah, penulis aktif menjadi administrator Laboratorium Manajemen Informasi Teknik Informatika dan aktif dalam berbagai organisasi baik tingkat jurusan maupun fakultas.

Dalam menyelesaikan pendidikan S1, penulis mengambil bidang minat Manajemen Informasi (MI) dan memiliki ketertarikan pada bidang *Web* dan *Mobile Application Development*. Sebagai mahasiswa, penulis berperan aktif dalam beberapa organisasi kampus seperti staf wahana dan budaya pada acara ITS EXPO 2014 dan menjadi staf ahli wahana dan budaya pada acara ITS EXPO 2015. Selain itu penulis juga menjadi staf Web dan 3D pada SCHEMATICS 2014 dan menjadi BPH 3D SCHEMATICS 2015, dan juga staf dan staf ahli minat bakat Himpunan Mahasiswa Teknik Computer-Informatika (HMTC) 2014-2015 dan 2015-2016. Penulis dapat dihubungi melalui email: riska.adhita079@gmail.com atau melalui facebook: www.facebook.com/riska.adhita.9